https://karthik-iitm.github.io/mlt-workshop/OPPE/nb-sample.html provide solution of this

Here are **step-by-step solutions and explanations** for the sample MLT OPPE shown:

Question 1

> The matrix `M` is of shape `(m, n)`. Enter $3m - 2n$ as your answer.

Code:

```
# DATA CELL
rng = np.random.default_rng(seed=1001)
m, k, n = rng.integers(100, 1000, 3)
A = rng.integers(0, 5, (m, k))
B = rng.integers(0, 5, (k, n))
M = A @ B

# SOLUTION
ans = 3 * m - 2 * n
print(ans)
```

Question 2

> Matrix `M` is of shape `(n, n)`. Find the dot product of the 230th row of M and the 158th column of M (1-based indexing).

Code:

```
# DATA CELL
rng = np.random.default_rng(seed=1001)
n = rng.integers(100, 300)
M = rng.integers(0, 5, (n, n))

# SOLUTION
row = M[230 - 1, :] # 229th (0-based)
col = M[:, 158 - 1] # 157th (0-based)
ans = np.dot(row, col)
print(ans)
```

Question 3

> Solve for x in $x\sin(x) = e^x$, $-1 \leq x \leq 0$. Use matplotlib to find (graphical root).

Code:

```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(-1, 0, 1000)
y1 = x * np.sin(x)
y2 = np.exp(x)
plt.plot(x, y1, label='x*sin(x)')
plt.plot(x, y2, label='exp(x)')
plt.legend()
plt.grid()
plt.show()

# Numeric solution
from scipy.optimize import bisect
def f(x):
    return x * np.sin(x) - np.exp(x)

root = bisect(f, -1, 0)
print(round(root, 3))
```

Question 4

> A perceptron model with weight `w` and test point `x_test`, label is sign(w @ x_test), output in {1, -1}

Code:

```
label = 1 if np.dot(w, x_test) >= 0 else -1
print(label)
```

Question 5

> Use perceptron to check linear separability (code terminates in ≤20 epochs means separable).

Code:

```
def perceptron(X, y, epochs=20):
    d, n = X.shape
    w = np.zeros(d)
    for epoch in range(epochs):
        changes = 0
        for i in range(n):
            if y[i] * np.dot(w, X[:, i]) <= 0:
```

```
                w += y[i] * X[:, i]
                changes += 1
        if changes == 0:
            return True
    return False

print("Dataset 1 separable?", perceptron(X_1, y_1))
print("Dataset 2 separable?", perceptron(X_2, y_2))
```

Questions 6-9: K-Means

Implement K-means, get final centers and answer:

```
# K-Means algorithm
def kmeans(X, centers):
    while True:
        # Assign clusters
        d, n = X.shape
        k = len(centers)
        labels = np.zeros(n, dtype=int)
        for i in range(n):
            labels[i] = np.argmin([np.linalg.norm(X[:, i] - c) for c in centers])
        # Compute new centers
        new_centers = []
        for j in range(k):
            if np.sum(labels == j) > 0:
                new_centers.append(X[:, labels == j].mean(axis=1))
            else: # If no points, keep old center
                new_centers.append(centers[j])
        # Check convergence
        if np.allclose(new_centers, centers):
            break
        centers = new_centers
    return centers, labels

centers0 = [c_1, c_2, c_3]
centers_final, labels = kmeans(X, centers0)
c_1_final, c_2_final, c_3_final = centers_final

# Q6-Q8
print(np.round(np.linalg.norm(c_1_final), 2)) # Q6
print(np.round(np.linalg.norm(c_2_final), 2)) # Q7
print(np.round(np.linalg.norm(c_3_final), 2)) # Q8

# Q9
test_cluster = np.argmin([np.linalg.norm(x_test - c) for c in centers_final])
print(test_cluster + 1)
```

Question 10 (SVM support vectors)

Count support vectors.

```
# Support vectors satisfy y_i*(w^* @ x_i) == 1
X = np.array([[2,2,0,1,1,0,-1,0,-1,0],[1,1,0,-1,-2,-3,0,-1,-1,-1,1,3,1,-1,1,1,-1,1,-1,1]]
y = np.array([-1,-1,-1,1,1,1,1,1])
w_star = np.array([-2,1])
n_sv = sum(np.isclose(y * (w_star @ X), 1))
print(n_sv)
```

Question 11 (SVM label for given point)

```
label = 1 if np.dot(w_star, np.array([2,1])) >= 0 else -1
print(label)
```

Questions 12-17: Linear & Ridge Regression

```
X_train = np.array([[1,0,1,-1,2],[2,1,0,-1,2],[0,1,2,3,1]])
y_train = np.array([1,2,0,3,1])
X_test = np.array([[1,0,2,1,4],[0,1,3,1,2],[-1,0,3,1,-1]])
y_test = np.array([-1,0,2,1,2])
d, n = X_train.shape

# Q12: How many weights?
print(d) # Each feature, no bias term

# Q13: Norm of weights (OLS)
w = np.linalg.pinv(X_train.T) @ y_train
print(np.round(np.linalg.norm(w), 2))

# Q14: RMSE train
y_pred_train = X_train.T @ w
rmse_train = np.sqrt(np.mean((y_pred_train - y_train)**2))
print(np.round(rmse_train, 3))

# Q15: RMSE test
y_pred_test = X_test.T @ w
rmse_test = np.sqrt(np.mean((y_pred_test - y_test)**2))
print(np.round(rmse_test, 2))

# Q16-17: Ridge for different lambdas
best_lambda_train = None
best_train_rmse = float('inf')
best_lambda_test = None
best_test_rmse = float('inf')

for lamb in [0.01, 0.1, 1, 10]:
    w_ridge = np.linalg.inv(X_train @ X_train.T + lamb * np.eye(d)) @ X_train @ y_train
    rmse_train = np.sqrt(np.mean((X_train.T @ w_ridge - y_train)**2))
    rmse_test = np.sqrt(np.mean((X_test.T @ w_ridge - y_test)**2))
```

```
    if rmse_train < best_train_rmse:
        best_train_rmse = rmse_train
        best_lambda_train = lamb
    if rmse_test < best_test_rmse:
        best_test_rmse = rmse_test
        best_lambda_test = lamb

print("Train: lambda with least error =", best_lambda_train) # Q16
print("Test: lambda with least error =", best_lambda_test)   # Q17
```

If you want **individual question explanations, outputs, or full step-by-step working** for any specific part, let me know!

⁂

1. https://karthik-iitm.github.io/mlt-workshop/OPPE/nb-sample.html