9. Using KNN algorithm for linear regression, get the fertilizer response for an agricultural experiment where the crop yield is tested against fertilizers. The response from crops is the variable.

import Libraries

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error
import numpy as np
```

sample dataset

```
data = [
    [20, 30, 40, 5.2],  # Fertilizer A - N in %, P in %, K in %, yield
in ton
    [25, 35, 45, 6.5],  # Fertilizer B - N in %, P in %, K in %, yield
in ton
    [15, 25, 35, 4.8],  # Fertilizer A - N in %, P in %, K in %, yield
in ton
    [30, 40, 50, 7.0],  # Fertilizer B - N in %, P in %, K in %, yield
in ton
    [10, 20, 30, 4.0],  # Fertilizer A - N in %, P in %, K in %, yield
in ton
    [28, 38, 48, 6.8],  # Fertilizer B - N in %, P in %, K in %, yield
in ton
]
```

separate features and response variable

```
x = [row[:-1] for row in data]
y = [row[-1] for row in data]
```

split the dataset into training and testing sets

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.2, random_state=42)
```

initialize KNN regressor

```
knn_regressor = KNeighborsRegressor(n_neighbors=3)
```

train the model

```
knn_regressor.fit(x_train, y_train)
```

```
KNeighborsRegressor(n_neighbors=3)
```

make predictions on the testing set

```
y_pred = knn_regressor.predict(x_test)
```

evaluate performance using mean squared error

```
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse:.2f}')
```

```
Mean Squared Error: 0.04
```

predict crop yield for new data

```
new_data = [[22, 32, 42], [18, 28, 38]]  # Replace with your actual
new data
predicted_yield = knn_regressor.predict(new_data)
print(f'Predicted Crop Yields: {predicted_yield}')
```

```
Predicted Crop Yields: [6.2 5.2]
```

Result

Using KNN algorithm for linear regression, a python code to get crop yield against fertilizer was developed and executed successfully.