ASSIGNMENT – 16

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, mean_squared_error

X_classification = np.random.rand(1000, 10)  # 1000 samples with 10 features
y_classification = np.random.randint(2, size=1000)  # Binary labels (0 or 1)
X_regression = np.random.rand(1000, 10)  # 1000 samples with 10 features
y_regression = np.random.rand(1000)  # Regression targets

# Split the data into training and testing sets
X_train_cls, X_test_cls, y_train_cls, y_test_cls = train_test_split(X_classification, y_classification,
test_size=0.2)
X_train_reg, X_test_reg, y_train_reg, y_test_reg = train_test_split(X_regression, y_regression,
test_size=0.2)

def create_and_train_model(X_train, y_train, X_test, y_test, task_type, activation_function, optimizer,
dropout_rate):
  model = keras.Sequential([
    keras.layers.Dense(64, activation=activation_function, input_shape=(10,)),
    keras.layers.Dropout(dropout_rate),
    keras.layers.Dense(32, activation=activation_function),
    keras.layers.Dropout(dropout_rate),
    keras.layers.Dense(1, activation='sigmoid' if task_type == 'classification' else 'linear')
  ])
  model.compile(optimizer=optimizer, loss='binary_crossentropy' if task_type == 'classification' else
'mean_squared_error')
  model.fit(X_train, y_train, epochs=10, batch_size=32, verbose=2)

  if task_type == 'classification':
    y_pred = (model.predict(X_test) > 0.5).astype(int)
    accuracy = accuracy_score(y_test, y_pred)
    print(f'Accuracy: {accuracy}')
  else:
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    print(f'Mean Squared Error: {mse}')
```

 # Binary Classification

```
print("Training Binary Classification Model...")
create_and_train_model(X_train_cls, y_train_cls, X_test_cls, y_test_cls, task_type='classification',
activation_function='relu', optimizer='adam', dropout_rate=0.5)
```

 # Regression

```
print("Training Regression Model...")
create_and_train_model(X_train_reg, y_train_reg, X_test_reg, y_test_reg, task_type='regression',
activation_function='tanh', optimizer='sgd', dropout_rate=0.3)
```