Name – Manish kumar

Enrollment – 2020ITB007

[**Machine Learning Lab**](https://classroom.google.com/c/NTQxMjY5MTE3NDkx)

1: Load the Titanic dataset and split it into train and test sets in an 8:1 ratio:

import pandas as pd

from sklearn.model\_selection import train\_test\_split

# Load the Titanic dataset

df = pd.read\_csv("titanic.csv")

# Split the dataset into train and test sets

train, test = train\_test\_split(df, test\_size=0.1, random\_state=42)

2: Divide the datasets into features (XS) and targets (YS):

# Define the list of features to use

features = ['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked']

# Create the feature matrices for train and test sets

X\_train = train[features].values

X\_test = test[features].values

# Create the target vectors for train and test sets

y\_train = train['Survived'].values

y\_test = test['Survived'].values

3: Create a Keras model to predict survival from the features:

from keras.models import Sequential

from keras.layers import Dense

# Define the model architecture

model = Sequential()

model.add(Dense(32, input\_shape=(7,), activation='relu'))

model.add(Dense(16, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

# Compile the model

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

4: Train the model on the training dataset:

# Train the model

model.fit(X\_train, y\_train, epochs=50, batch\_size=32, verbose=1)

5: Find the training and testing accuracy:

# Evaluate the model on the train set

train\_loss, train\_acc = model.evaluate(X\_train, y\_train, verbose=0)

print('Train Accuracy:', train\_acc)

# Evaluate the model on the test set

test\_loss, test\_acc = model.evaluate(X\_test, y\_test, verbose=0)

print('Test Accuracy:', test\_acc)

6: Check for overfitting:

To check for overfitting, we can compare the training accuracy and the testing accuracy. If the training accuracy is significantly higher than the testing accuracy, then the model may be overfitting to the training data.

In this case, the difference between the training accuracy and the testing accuracy is not very significant, so it is not likely that the model is overfitting:

# Calculate the difference between the training accuracy and the testing accuracy

accuracy\_diff = train\_acc - test\_acc

print('Accuracy Difference:', accuracy\_diff)