CodeAlpha Machine Learning Internship Tasks (Simplified)

# ✅ Task 1: Credit Scoring Model

🎯 Objective

Predict whether a person is creditworthy using their past financial data (like income, debts, payment history).

🧠 Concepts Used

* - Pandas for handling data
* - Scikit-learn for machine learning
* - Logistic Regression (simple classifier)
* - Evaluation metrics: Accuracy, Precision, Recall

💻 Python Code (Simple Version)

\*\*import pandas as pd\*\*  
\*\*from sklearn.model\_selection import train\_test\_split\*\*  
\*\*from sklearn.linear\_model import LogisticRegression\*\*  
\*\*from sklearn.metrics import accuracy\_score, classification\_report\*\*  
  
# Load dataset  
\*\*data = pd.read\_csv("credit\_data.csv")\*\*  
  
# Features and Target  
\*\*X = data.drop("Creditworthy", axis=1)\*\*  
\*\*y = data["Creditworthy"]\*\*  
  
# Split the data  
\*\*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=1)\*\*  
  
# Create and train model  
\*\*model = LogisticRegression()\*\*  
\*\*model.fit(X\_train, y\_train)\*\*  
  
# Predict  
\*\*predictions = model.predict(X\_test)\*\*  
  
# Results  
\*\*print("Accuracy:", accuracy\_score(y\_test, predictions))\*\*  
\*\*print(classification\_report(y\_test, predictions))\*\*

📝 Explanation

We used LogisticRegression() as it's simple and best for binary classification. Dataset must include numeric columns and a Creditworthy column (1 = yes, 0 = no). The code splits the data into training and testing sets, trains the model, and shows results.

📁 Suggested Folder Structure

CodeAlpha\_CreditScoringModel/  
├── credit\_scoring.py  
├── credit\_data.csv  
├── README.md

# ✅ Task 2: Disease Prediction from Medical Data

🎯 Objective

Use patient data (like age, symptoms, test results) to predict the risk of a disease (like diabetes or heart disease).

🧠 Concepts Used

* - CSV data handling
* - Decision Tree classifier
* - Accuracy and basic classification metrics

💻 Python Code (Simple Version)

\*\*import pandas as pd\*\*  
\*\*from sklearn.model\_selection import train\_test\_split\*\*  
\*\*from sklearn.tree import DecisionTreeClassifier\*\*  
\*\*from sklearn.metrics import accuracy\_score\*\*  
  
# Load data  
\*\*data = pd.read\_csv("disease\_data.csv")\*\*  
  
# Features and Target  
\*\*y = data["Disease"]\*\* # 1 for disease, 0 for no disease  
\*\*X = data.drop("Disease", axis=1)\*\*  
  
# Split data  
\*\*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=1)\*\*  
  
# Model  
\*\*model = DecisionTreeClassifier()\*\*  
\*\*model.fit(X\_train, y\_train)\*\*  
  
# Predict  
\*\*pred = model.predict(X\_test)\*\*  
\*\*print("Accuracy:", accuracy\_score(y\_test, pred))\*\*

📝 Explanation

We use a decision tree because it's easy to understand and works well with medical data. The dataset should have a Disease column with 0 or 1 values. Features can include age, glucose level, etc.

📁 Suggested Folder Structure

CodeAlpha\_DiseasePrediction/  
├── disease\_predictor.py  
├── disease\_data.csv  
├── README.md

# ✅ Task 3: Handwritten Character Recognition

🎯 Objective

Recognize handwritten digits (0–9) using image data and a basic neural network.

🧠 Concepts Used

* - Image data (MNIST dataset)
* - Keras (a high-level deep learning library)
* - CNN (Convolutional Neural Network)

💻 Python Code (Simple Version)

\*\*import tensorflow as tf\*\*  
\*\*from tensorflow.keras.datasets import mnist\*\*  
\*\*from tensorflow.keras.models import Sequential\*\*  
\*\*from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense\*\*  
\*\*from tensorflow.keras.utils import to\_categorical\*\*  
  
# Load dataset  
\*\*(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()\*\*  
  
# Reshape and normalize  
\*\*x\_train = x\_train.reshape(-1, 28, 28, 1) / 255.0\*\*  
\*\*x\_test = x\_test.reshape(-1, 28, 28, 1) / 255.0\*\*  
  
# One-hot encoding  
\*\*y\_train = to\_categorical(y\_train)\*\*  
\*\*y\_test = to\_categorical(y\_test)\*\*  
  
# Build model  
\*\*model = Sequential([  
 Conv2D(32, (3, 3), activation='relu', input\_shape=(28, 28, 1)),  
 MaxPooling2D(2, 2),  
 Flatten(),  
 Dense(64, activation='relu'),  
 Dense(10, activation='softmax')  
])\*\*  
  
\*\*model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])\*\*  
\*\*model.fit(x\_train, y\_train, epochs=5, validation\_data=(x\_test, y\_test))\*\*

📝 Explanation

We use the MNIST dataset which is built-in in Keras. The CNN architecture is kept simple: one conv layer, pooling, flatten, and two dense layers. It trains in just a few minutes on a basic laptop.

📁 Suggested Folder Structure

CodeAlpha\_HandwrittenRecognition/  
├── digit\_recognizer.py  
├── README.md