CodeAlpha Internship Tasks Summary

Task 1

```python

# Import libraries

import pandas as pd import seaborn as sns import matplotlib.pyplot as plt from sklearn.modelselection import traintestsplit from sklearn.ensemble import RandomForestClassifier from sklearn.metrics import accuracyscore, classificationreport, confusionmatrix import os

# Load dataset

os.chdir(r"C:\Users\Sourav\CodeAlphaproject") df = pd.readcsv("Iris.csv")

# Display basic info

print(df.head()) print(df.info())

```

```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Speci
0   1            5.1           3.5            1.4           0.2  Iris-seto
1   2            4.9           3.0            1.4           0.2  Iris-seto
2   3            4.7           3.2            1.3           0.2  Iris-seto
3   4            4.6           3.1            1.5           0.2  Iris-seto
4   5            5.0           3.6            1.4           0.2  Iris-seto
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
None
```

```python

# Visualize data

sns.pairplot(df, hue="Species") plt.show() ```

png

```python

# Prepare data

X = df.drop("Species", axis=1) y = df["Species"]

# Split into train and test

Xtrain, Xtest, ytrain, ytest = traintestsplit(X, y, testsize=0.2, randomstate=42)

# Train model

model = RandomForestClassifier() model.fit(Xtrain, ytrain)

# Predict and evaluate

ypred = model.predict(Xtest) print("Accuracy:", accuracyscore(ytest, ypred)) print(classificationreport(ytest, ypred))

```

```
Accuracy: 1.0
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      1.00      1.00         9
 Iris-virginica       1.00      1.00      1.00        11

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```

```python

```
```

Task 2

```python

# Import libraries

import pandas as pd import numpy as np import seaborn as sns import matplotlib.pyplot as plt from sklearn.modelselection import traintestsplit from sklearn.linearmodel import LinearRegression from sklearn.metrics import meansquarederror, r2_score import os

# Load dataset

os.chdir(r"C:\Users\Sourav\CodeAlpha_project")

df = pd.read_csv("car data.csv")

# View the dataset

print(df.head()) print(df.info())

```

```
  Car_Name  Year  Selling_Price  Present_Price  Driven_kms Fuel_Type  \
0     ritz  2014           3.35           5.59       27000    Petrol
1      sx4  2013           4.75           9.54       43000    Diesel
2     ciaz  2017           7.25           9.85        6900    Petrol
3  wagon r  2011           2.85           4.15        5200    Petrol
4    swift  2014           4.60           6.87       42450    Diesel

  Selling_type Transmission  Owner
0       Dealer       Manual      0
1       Dealer       Manual      0
2       Dealer       Manual      0
3       Dealer       Manual      0
4       Dealer       Manual      0
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Car_Name       301 non-null    object
 1   Year           301 non-null    int64
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Driven_kms     301 non-null    int64
 5   Fuel_Type      301 non-null    object
 6   Selling_type   301 non-null    object
```

```
 7   Transmission   301 non-null    object
 8   Owner          301 non-null    int64
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
None
```

```python
# Drop car name

df = df.drop("Car_Name", axis=1)

# Check categorical features

print(df.select_dtypes(include='object').columns)

# Encode categorical variables

df = pd.getdummies(df, dropfirst=True)

# Features and target

X = df.drop("SellingPrice", axis=1) y = df["SellingPrice"]

# Split into train and test

Xtrain, Xtest, ytrain, ytest = traintestsplit(X, y, testsize=0.2,
randomstate=42)

# Train model

model = LinearRegression() model.fit(Xtrain, ytrain)

# Predict

ypred = model.predict(Xtest)

# Evaluation

rmse = np.sqrt(meansquarederror(ytest, ypred)) r2 = r2score(ytest, y_pred)
```

print("Root Mean Squared Error:", round(rmse, 2)) print("R² Score:", round(r2, 2))

Actual vs Predicted plot

plt.figure(figsize=(8, 6)) sns.scatterplot(x=ytest, y=ypred) plt.xlabel("Actual Price") plt.ylabel("Predicted Price") plt.title("Actual vs Predicted Car Prices") plt.grid(True) plt.show()

```

```
Index(['Fuel_Type', 'Selling_type', 'Transmission'], dtype='object')
Root Mean Squared Error: 1.87
R² Score: 0.85
```

png

```python
```

Task 3

```python

# Import libraries

import pandas as pd import seaborn as sns import matplotlib.pyplot as plt from sklearn.linearmodel import LinearRegression from sklearn.metrics import meansquarederror, r2score from sklearn.modelselection import traintest_split import numpy as np import os

# Load dataset

os.chdir(r"C:\Users\Sourav\CodeAlpha_project")

df = pd.read_csv('Advertising.csv')

# View data

print(df.head()) print(df.info()) ```
```

```
   Unnamed: 0     TV  Radio  Newspaper  Sales
0           1  230.1   37.8       69.2   22.1
1           2   44.5   39.3       45.1   10.4
2           3   17.2   45.9       69.3    9.3
3           4  151.5   41.3       58.5   18.5
```
```

```
4           5  180.8   10.8       58.4   12.9
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  200 non-null    int64
 1   TV          200 non-null    float64
 2   Radio       200 non-null    float64
 3   Newspaper   200 non-null    float64
 4   Sales       200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
None
```

```python

# Visualize correlations

sns.pairplot(df, xvars=['TV', 'Radio', 'Newspaper'], yvars='Sales', kind='reg')
plt.show()

# Check correlation

print(df.corr())

# Features and Target

X = df[['TV', 'Radio', 'Newspaper']] y = df['Sales']

# Train-Test Split

Xtrain, Xtest, ytrain, ytest = traintestsplit(X, y, testsize=0.2,
randomstate=42)

# Train model

model = LinearRegression() model.fit(Xtrain, ytrain)

# Predict

ypred = model.predict(Xtest)
```

Evaluate

rmse = np.sqrt(meansquarederror(ytest, ypred)) r2 = r2score(ytest, y_pred)

print("Root Mean Squared Error:", round(rmse, 2)) print("R² Score:", round(r2, 2))

Visualize prediction

plt.figure(figsize=(8, 6)) sns.scatterplot(x=ytest, y=ypred) plt.xlabel("Actual Sales") plt.ylabel("Predicted Sales") plt.title("Actual vs Predicted Sales") plt.grid(True) plt.show()

```

png

```
             Unnamed: 0   TV   Radio Newspaper   Sales
Unnamed: 0   1.000000 0.017715 -0.110680 -0.154944 -0.051616
TV   0.017715 1.000000 0.054809   0.056648 0.782224
Radio   -0.110680 0.054809 1.000000   0.354104 0.576223
Newspaper   -0.154944 0.056648 0.354104   1.000000 0.228299
Sales   -0.051616 0.782224 0.576223   0.228299 1.000000
Root Mean Squared Error: 1.78
R² Score: 0.9
```

png

```python

```
```