```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score


from google.colab import files
# Use files.upload() to prompt the user to upload files
uploaded = files.upload()
data = pd.read_csv(next(iter(uploaded)))  # e.g., "companies.csv"
print(data.head())
```

```
Choose Files 50_Startup…mpanies.csv
• 50_Startups_companies.csv(text/csv) - 1964 bytes, last modified: 6/10/2025 - 100% done
Saving 50_Startups_companies.csv to 50_Startups_companies.csv
   R&D Spend  Administration  Marketing Spend     Profit
0  165349.20       136897.80        471784.10  192261.83
1  162597.70       151377.59        443898.53  191792.06
2  153441.51       101145.55        407934.54  191050.39
3  144372.41       118671.85        383199.62  182901.99
4  142107.34        91391.77        366168.42  166187.94
```

```
features = ['R&D Spend', 'Administration', 'Marketing Spend']
target = 'Profit'

X = data[features]
y = data[target]

display(X.head())
display(y.head())
```

	R&D Spend	Administration	Marketing Spend
0	165349.20	136897.80	471784.10
1	162597.70	151377.59	443898.53
2	153441.51	101145.55	407934.54
3	144372.41	118671.85	383199.62
4	142107.34	91391.77	366168.42

	Profit
0	192261.83
1	191792.06
2	191050.39
3	182901.99
4	166187.94

dtype: float64

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

display(X_train.head())
display(X_test.head())
display(y_train.head())
display(y_test.head())
```

	R&D Spend	Administration	Marketing Spend
12	93863.75	127320.38	249839.44
4	142107.34	91391.77	366168.42
37	44069.95	51283.14	197029.42
8	120542.52	148718.95	311613.29
3	144372.41	118671.85	383199.62

	R&D Spend	Administration	Marketing Spend
13	91992.39	135495.07	252664.93
39	38558.51	82982.09	174999.30
30	61994.48	115641.28	91131.24
45	1000.23	124153.04	1903.93
17	94657.16	145077.58	282574.31

	Profit
12	141585.52
4	166187.94
37	89949.14
8	152211.77
3	182901.99

dtype: float64

	Profit
13	134307.35
39	81005.76
30	99937.59
45	64926.08
17	125370.37

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Model-1: Linear Regression

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np

lr = LinearRegression()
lr.fit(X_train_scaled, y_train)

y_pred_lr = lr.predict(X_test_scaled)

print("Linear Regression Metrics:")
print(f"MAE: {mean_absolute_error(y_test, y_pred_lr):.2f}")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test, y_pred_lr)):.2f}")
print(f"R²: {r2_score(y_test, y_pred_lr):.4f}")
```

```
Linear Regression Metrics:
MAE: 6979.15
RMSE: 8995.91
R²: 0.9001
```

Model 2: Decision Tree Regression

```
from sklearn.tree import DecisionTreeRegressor

dt = DecisionTreeRegressor(random_state=42)
dt.fit(X_train_scaled, y_train)

y_pred_dt = dt.predict(X_test_scaled)
```

```
print("Decision Tree Metrics:")
print(f"MAE: {mean_absolute_error(y_test, y_pred_dt):.2f}")
print(f"RMSE: {mean_squared_error(y_test, y_pred_dt):.2f}")
print(f"R²: {r2_score(y_test, y_pred_dt):.4f}")
```

```
Decision Tree Metrics:
MAE: 13755.66
RMSE: 400026479.25
R²: 0.5060
```

Model 3: Random Forest Regression

```
from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor(random_state=42)
rf.fit(X_train_scaled, y_train)

y_pred_rf = rf.predict(X_test_scaled)

print("Random Forest Metrics:")
print(f"MAE: {mean_absolute_error(y_test, y_pred_rf):.2f}")
print(f"RMSE: {mean_squared_error(y_test, y_pred_rf):.2f}")
print(f"R²: {r2_score(y_test, y_pred_rf):.4f}")
```

```
Random Forest Metrics:
MAE: 6437.50
RMSE: 72625008.62
R²: 0.9103
```

Model 4: Support Vector Regression (SVR)

```
from sklearn.svm import SVR

svr = SVR()
svr.fit(X_train_scaled, y_train)

y_pred_svr = svr.predict(X_test_scaled)

print("SVR Metrics:")
print(f"MAE: {mean_absolute_error(y_test, y_pred_svr):.2f}")
print(f"RMSE: {mean_squared_error(y_test, y_pred_svr):.2f}")
print(f"R²: {r2_score(y_test, y_pred_svr):.4f}")
```

```
SVR Metrics:
MAE: 22846.73
RMSE: 955620367.28
R²: -0.1801
```

```
results = {
    "Model": ["Linear Regression", "Decision Tree", "Random Forest", "SVR"],
    "MAE": [6979.15, 13755.66, 6437.50, 22846.73],
    "RMSE": [8995.91, 400026479.25, 72625008.62, 955620367.28],
    "R²": [0.9001, 0.5060, 0.9103, -0.1801]
}

results_df = pd.DataFrame(results)
print("Model Comparison:")
print(results_df.sort_values(by="R²", ascending=False))
```

```
Model Comparison:
                Model       MAE          RMSE      R²
2      Random Forest   6437.50  7.262501e+07  0.9103
0  Linear Regression   6979.15  8.995910e+03  0.9001
1      Decision Tree  13755.66  4.000265e+08  0.5060
3                SVR  22846.73  9.556204e+08 -0.1801
```

```
best_model = max(results_df.to_dict('records'), key=lambda x: x['R²'])
print(f"Best Model: {best_model['Model']} (R² = {best_model['R²']:.4f})")
```

```
Best Model: Random Forest (R² = 0.9103)
```

```
import pandas as pd
import sklearn
import numpy as np
import xgboost
import matplotlib

print(f"pandas: {pd.__version__}")
print(f"scikit-learn: {sklearn.__version__}")
```

```
print(f"numpy: {np.__version__}")
print(f"xgboost: {xgboost.__version__}")
print(f"matplotlib: {matplotlib.__version__}")
```

```
pandas: 2.2.2
scikit-learn: 1.6.1
numpy: 2.0.2
xgboost: 2.1.4
matplotlib: 3.10.0
```