Predictive_Models

Predict Sales using Linear Regression

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error

X = df[['Quantity','Discount']]
y = df['Sales']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

reg = LinearRegression().fit(X_train, y_train)
y_pred = reg.predict(X_test)

print("R² Score:", r2_score(y_test, y_pred))
print("RMSE:", mean_squared_error(y_test, y_pred, squared=False))
```

```
R² Score: -0.013869503401514605
```

Predict Profit using Multiple Regression

```
X = df[['Sales','Quantity','Discount']]
y = df['Profit']

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

model = LinearRegression().fit(X_train, y_train)
print("R² Score:", model.score(X_test, y_test))
```

```
R² Score: 0.8075936018035006
```

Predict Profitability using Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix

tree = DecisionTreeClassifier(max_depth=5, random_state=42).fit(X_train, y_train)
y_pred = tree.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy: 1.0
Confusion Matrix:
 [[200]]
```

Predict Profitability using Random Forest

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=100, random_state=42).fit(X_train, y_train)
print("Accuracy:", accuracy_score(y_test, rf.predict(X_test)))
```

```
Accuracy: 1.0
```

Predict Profitability using KNN (K-Nearest Neighbors)

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=5).fit(X_train, y_train)
print("Accuracy:", accuracy_score(y_test, knn.predict(X_test)))
```

```
Accuracy: 1.0
```

Predict Profit using Polynomial Regression

```
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline

poly_model = make_pipeline(PolynomialFeatures(degree=2), LinearRegression())
poly_model.fit(X_train[['Discount']], y_train)

y_pred = poly_model.predict(X_test[['Discount']])
print("R² Score:", r2_score(y_test, y_pred))
```

```
R² Score: 1.0
```

Predict Profit using Ridge Regression (Regularization)

```
from sklearn.linear_model import Ridge

ridge = Ridge(alpha=1.0).fit(X_train, y_train)
print("R² Score (Ridge):", ridge.score(X_test, y_test))
```

```
R² Score (Ridge): 1.0
```

Predict Profit using Lasso Regression

```
from sklearn.linear_model import Lasso

lasso = Lasso(alpha=0.01).fit(X_train, y_train)
print("R² Score (Lasso):", lasso.score(X_test, y_test))
```

```
R² Score (Lasso): 1.0
```