PRODUCT SALES ANALYSIS

PHASE2: INNOVATION

SUBMMITTED BY

NAME :THATHIREDDY JYOTHIREDDY
NM ID : au723921244050
MAIL ID :jyothireddythathireddy.com

PHASE2: INNOVATION

Incorporating Machine learning algorithm for predicting future sales trends or customer behaviours.

There are several machine learning algorithms that can be used for predicting future sales trends or customer behaviours. The choice of algorithm depends on factors like the nature of the data, the complexity of the problem, and the size of the dataset. Here are some commonly used algorithms for this purpose:

Linear Regression:

Linear regression models the relationship between a dependent variable (sales or customer behaviour) and one or more independent variables (features like time, marketing spend, etc.). It's useful when there's a linear relationship between the variables.

Use Case: Predicting sales based on one or more continuous variables (e.g., advertising spend, time, etc.).

Strengths: Simple and interpretable, works well when there is a linear relationship between input variables and sales.

Linear Regression Algorithm:

1.Data Collection: Using the data from given dataset.

2.Data Pre-processing: Clean and pre-process the data. This may involve handling missing values, encoding categorical variables, and scaling numerical features if necessary.
3.Data Splitting: Split the data into two sets: a training set and a testing set. The training set is used to train the model, and the testing set is used to evaluate its performance.

4.Linear Regression Model: Train a linear regression model on the training data. The model will learn the coefficients (weights) for each independent variable to predict the dependent variable. The model's equation typically looks like this:

y = b0 + b1*x1 + b2*x2 + ... + bn*xn

y is the dependent variable (e.g., future sales).

b0 is the intercept (bias) term.

b1, b2, ... bn are the coefficients for the independent variables x1, x2, ... xn.

5.Make Predictions: Once satisfied with the model's performance, you can use it to make predictions on new data, such as future time periods, to predict sales trends or customer behavior.

6.Monitor and Refine: Continuously monitor the model's performance and update it as new data becomes available. This will help you maintain its accuracy as customer behaviours and sales trends evolve over time.

Random Forests:

Random Forests are an ensemble of decision trees that can handle more complex relationships. They're good for handling both categorical and numerical data.

Use Case: Predicting sales with a large number of features or variables.

Strengths: Handles non-linearity, can capture complex relationships, and is less prone to overfitting.

Random Forests algorithm:

1.Data Collection: Using the data from given dataset.

2.Data Preprocessing: Clean and preprocess the data. This may involve handling missing values, encoding categorical variables, and scaling numerical features if necessary.

3.Data Splitting: Split the data into two sets: a training set and a testing set. The training set is used to train the Random Forest model, and the testing set is used to evaluate its performance.

4.Random Forest Model: Train a Random Forest model on the training data. The model will consist of multiple decision trees, each trained on a random subset of the data with replacement (bootstrap sampling). Additionally, at each split in the tree, only a random subset of features is considered. This randomness helps to reduce overfitting.

5.Model Evaluation: Evaluate the model's performance on the testing set using appropriate metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), or R-squared (R2). These metrics will help you assess how well the model predicts future sales or customer behavior.

6.Make Predictions: Once you are satisfied with the model's performance, you can use it to make predictions on new data, such as future time periods, to predict sales trends or customer behavior.

7.Monitor and Refine: Continuously monitor the model's performance and update it as new data becomes available. This will help you maintain its accuracy as customer behavior and sales trends evolve over time.

Support Vector Machines (SVM):

SVM tries to find the best hyperplane that separates classes in a high-dimensional feature space. It can be used for regression (SVR) as well.\

Use Case: Classification problems related to customer behavior (e.g., churn prediction).

Strengths: Effective for high-dimensional spaces and works well when there is a clear margin of separation.

Support Vector Machines (SVM) algorithm:

1. Data Preparation: Obtain and pre-process the dataset

2. Splitting the Dataset: Split the dataset into training and testing sets:

3. Feature Selection/Extraction: Identify relevant features: Analyze the dataset and select the features that are most informative for the problem you are trying to solve

4. Model Training:

- Choose an appropriate SVM implementation.
- Set hyperparameters.
- Fit the model.

5. Model Evaluation:

- Predict on the testing set
- Evaluate model performance

6. Hyperparameter Tuning:

- Optimize hyperparameters: If the initial model performance is not satisfactory, consider performing hyperparameter tuning to find the optimal combination of hyperparameters

7. Final Model Deployment:

- Once satisfied with the model's performance, deploy it for real-world applications by using it to predict new, unseen instances.

K-Nearest Neighbours (KNN):

KNN predicts the target value for a new data point by averaging the values of its k-nearest neighbours. It's simple and can be effective for certain types of data.

Use Case: Clustering similar customer behavior or predicting based on similar sales patterns.

Strengths: Simple and intuitive, works well with small to medium-sized datasets.

K-Nearest Neighbours (KNN) algorithm:

1.Data Collection and Pre-processing: Obtain and pre-process the dataset

2.Feature Selection: Identify the most relevant features for predicting sales. This can be done through techniques like feature importance analysis or domain knowledge.

3.Split Data: Split the data into training and testing sets. The training set will be used to train the KNN model, while the testing set will be used to evaluate its performance.

4.Choose the Value of K: K in KNN represents the number of nearest neighbours to consider. Choose an appropriate value of K, which can be determined through cross-validation or other validation techniques.

5.Distance Metric: Select a distance metric (e.g., Euclidean distance, Manhattan distance) to measure the similarity between data points. The choice of distance metric can impact the results.

6.Model Training: Train the KNN model on the training data. The model stores the entire dataset in memory and uses it to make predictions.

7.Model Evaluation: Use the testing set to evaluate the model's performance. Common regression metrics include Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R2) score.

8.Predict Future Sales: Once the model is trained and evaluated, you can use it to predict future sales trends. Prepare a dataset with the relevant features for the future time period you're interested in.

9.Interpretation: Analyze the model to understand which features are contributing the most to the sales predictions. This can provide valuable insights into the factors driving sales trends.