## Project Title: Future Sales Prediction

**Phase 5: Documentation**

**Problem Definition and Design Thinking**

**Problem Definition:**

The problem is to develop a predictive model that uses historical sales data to forecast future sales for a retail company. The objective is to create a tool that enables the company to optimize inventory management and make informed business decisions based on datadriven sales predictions. This project involves data preprocessing, feature engineering, model selection, training, and evaluation.

**Design Thinking:**

**Data Source:**

Utilize a dataset containing historical sales data, including features like date, product ID, store ID, and sales quantity.

**Data Preprocessing:**

Clean and preprocess the data, handle missing values, and convert categorical features into numerical representations.

**Feature Engineering:**

Create additional features that could enhance the predictive power of the model, such as time-based features (e.g., day of the week, month).

**Model Selection:**

Choose suitable time series forecasting algorithms (e.g., ARIMA, Exponential Smoothing) for predicting future sales.

**Model Training:**

Train the selected model using the preprocessed data.

**Evaluation:**

Evaluate the model's performance using appropriate time series forecasting metrics (e.g., Mean Absolute Error, Root Mean Squared Error).

**Innovation**

**Description:** Consider exploring more advanced time series forecasting techniques like Prophet or LSTM networks for improved accuracy in predicting future sales.

Implementing advanced time series forecasting techniques like Prophet or LSTM networks can indeed improve the accuracy of predicting future sales.

**Here's a brief overview of each approach:**

**Prophet:**

Prophet is an open-source forecasting tool developed by Facebook that is specifically designed to handle time series data with strong seasonal effects and missing data. It can handle daily observations with seasonal patterns, holidays, and special events. Utilizing Prophet's capabilities can provide more accurate forecasts for your sales data, especially if there are distinct patterns or seasonal variations.

**LSTM Networks:**

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) that can model long-term dependencies in time series data. LSTMs are powerful for time series forecasting due to their ability to capture complex patterns and relationships in sequential data. They can be trained to understand the sales data's historical patterns and make predictions for future sales.

To implement these approaches for sales prediction, you would typically follow these steps:

**1. Data Preprocessing:**

Prepare your sales data for modeling by cleaning, transforming, and organizing it into a suitable format for the chosen technique.

**2. Model Selection and Training:**

Select either Prophet or LSTM based on the nature of your data and problem. Train the chosen model using historical sales data, tuning the hyperparameters for optimal performance.

**3. Validation and Testing:**

Split your dataset into training and testing sets to validate the model's performance. Adjust the model and fine-tune parameters as needed to improve accuracy.

**4. Forecasting:**

Use the trained model to make predictions for future sales based on the patterns learned from the historical data.

**Developement Part 1**

**Description :** Begin building the future sales prediction model by loading and preprocessing the dataset. Load the historical sales dataset and preprocess the data for analysis.

**Dataset Link** https://www.kaggle.com/datasets/chakradharmattapalli/future-sales-prediction

**Procedure :**

Following steps are used to load and preprocess the dataset for “Future Sales Prediction”.

**Step 1 :**

Import the python libraries with below code.

Import pandas as pd

**Step 2 :**

Load the dataset with below code.

FSP\_1=pd.read\_csv(‘https://www.kaggle.com/datasets/chakradharmattapalli/future-sales-prediction/download’)

Output :

0 TV Radio Newspaper Sales

1 230.1 37.8 69.2 22.1

2 44.5 39.3 45.1 10.4

3 17.2 45.9 69.3 12

4 151.5 41.3 58.5 16.5

… … … … … … … …

199 177 9.3 6.4 14.8

200 283.6 42 66.2 25.5

201 232.1 8.6 8.7 18.4

[201 rows x 4 columns]

**Step 3:**

Print the basic information about the dataset with below code.

FSP\_1.info()

Output :

Data columns (total 4 columns):

# Column Non-Null Count Dtype

— —------ —----------- —----

0 TV non-null 201 int64

1 Radio non-null 201 int64

2 NewsPapernon-null 201 int64

3 Sales non-null 201 int64

Dtypes: int64(4)

Memory usage:6.0+ KB

**Step 4:**

Preprocess the dataset

Check missing values with below code

FSP\_1.isnull().sum()

Output :

TV 0

Radio 0

Newspaper 0

Sales 0

Dtype:int64

Convert categorical features into numerical features

Define a function to convert categorical features into numerical features.

def encode\_categorical\_feature(df,column):

Return pd.get\_dummies(df[column,drop\_first=True)

Encode the TV value.

FSP\_2=encode\_categorical\_feature(FSP\_1,’TV’)

Encode the Radio value.

FSP\_2=encode\_categorical\_feature(FSP\_1,’Radio’)

Output :

TV Radio NewsPaper Sales

0 69.2 22.1

**Step 5 :**

Scale the numerical feature.

Define a function to scale numerical feature.

from sklearn.preprocessing import StandardScaler

Def scale\_numerical\_feature(df,columns):

scaler=StandardScaler()

Scaled\_df=scaler.fit\_transform(df[columns])

Return scaled\_df

Scale the numerical feature

Numerical\_feature=[‘Sales’]

FSP\_2=pd.concat([FSP\_1,scale\_numerical\_feature(FSP\_1,numerical\_feature)],axis=1)

**Step 6 :**

Split the dataset into training and test sets with below code.

From sklearn.model\_selection import train\_test\_split

X=FSP\_1.drop(‘TV’,axis=1)

Y=FSP\_1[‘TV’]

X\_train,X\_test,y\_train,y\_test=train\_test\_split(X,y,test\_size=0.25,random\_state=42)

Output :

(156,46)

This means that the training set contains 156 samples and test set contains 46 samples.

We have now loaded and preprocesed the future sales prediction dataset.

**Development Part 2**

**Description:**

Continue building the "Future Sales prediction" model by:

* Feature engineering
* Model training
* Evaluation.

Input:

import pandas as pd

import numpy as no

import matplotlib.pyplot as plt

import seaborn as sns

import plotly.express as px

from datetime import datetime,timedelta

data = pd.read\_csv("C://Users/Admin/Documents/Phase4/Sales\_data.csv",encoding = "ISO-8859-1")

data\_date = data.copy()

data.head(5)

Output:

0 TV Radio Newspaper Sales

1 230.1 37.8 69.2 22.1

2 44.5 39.3 45.1 10.4

3 17.2 45.9 69.3 12

4 151.5 41.3 58.5 16.5

**Feature Engineering:**

Feature engineering is the process of creating new features from existing ones, or transforming existing features in a way that makes them more informative for the machine learning model.

We'll create a new feature called "Total Advertising" by summing the expenses from TV, Newspaper, Radio and Sales.

Input:

import pandas as pd

data = {pd.read\_csv("C://Users/Admin/Documents/Phase4/Sales\_data.csv",encoding = "ISO-8859-1")

}

data\_date = data.copy()

df = pd.DataFrame(data)

df['Total Advertising'] = df['TV'] + df['Newspaper'] + df['Radio']

print(df)

Output:

TV Radio Newspaper Sales Total Advertising

0 230.1 37.8 69.2 22.1 337.1

1 44.5 39.3 45.1 10.4 128.9

2 17.2 45.9 69.3 12.0 132.4

3 151.5 41.3 58.5 16.5 251.3

4 180.8 10.8 58.4 17.9 250.0

…. …. …. ….

**Model Training:**

Model training in machine learning refers to the process of using a dataset, typically with input features (independent variables) and corresponding target values (dependent variable), to teach a machine learning model to make predictions or decisions.

Input:

import pandas as pd

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

data = { pd.read\_csv("C://Users/Admin/Documents/Phase4/Sales\_data.csv",encoding = "ISO-8859-1")

}

data\_date = data.copy()

df = pd.DataFrame(data)

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

y = df['Sales']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = LinearRegression()

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

print("Coefficients:", model.coef\_)

print("Intercept:", model.intercept\_)

print("Predicted Sales:")

print(y\_pred)

Output:

Coefficients: [0.04555832 0.18878124 0.1897103]

Intercept: 2.979067908422406

Predicted Sales:

[12.5181044 13.10443243]

**Evaluation:**

In machine learning, evaluation refers to the process of assessing the performance and accuracy of a trained model on a dataset. It helps determine how well the model generalizes to new, unseen data. Common evaluation metrics include Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R2).

Input:

import pandas as pd

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score

data = {pd.read\_csv("C://Users/Admin/Documents/Phase4/Sales\_data.csv",encoding = "ISO-8859-1")

}

df = pd.DataFrame(data)

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

y = df['Sales']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = LinearRegression()

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)

mae = mean\_absolute\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error (MSE):", mse)

print("Root Mean Squared Error (RMSE):", rmse)

print("Mean Absolute Error (MAE):", mae)

print("R-squared (R2):", r2)

Output:

Mean Squared Error (MSE): 4.233428596014785

Root Mean Squared Error (RMSE): 2.0573745712091754

Mean Absolute Error (MAE): 1.2725084639076798

R-squared (R2): 0.8776191512296577

We have to clearly outline the problem statement, design thinking process, and the phases of development.Describe the dataset used, data preprocessing steps, and model training process.Explain the choice of evaluation metrics.

In the above process we had tried our best.

Thank you