APPLIED DATA SCIENCE

Sakthiswathi g
III-cse
vvcet

Product Demand Prediction with Machine Learnings

ABSTRACTION:

1. Data Collection: Collect historical sales data and external factors that influence demand, such as marketing campaigns, holidays, economic indicators, etc.
2. Data Preprocessing: Clean and preprocess the data, handle missing values, and convert categorical features into numerical representations
3. Feature Engineering: Create additional features that capture seasonal patterns, trends, and external influences on product demand.
4. Model Selection: Choose suitable regression algorithms (e.g., Linear Regression, Random Forest, XGBoost) for demand forecasting.
5. Model Training: Train the selected model using the preprocessed data.
6. Evaluation: Evaluate the model's performance using appropriate regression metrics (e.g., Mean Absolute Error, Root Mean Squared Error).

Data Collection

<u>a. Historical Sales Data:</u>

- Extract sales data from your internal sources. Ensure that it is clean, accurate, and well-documented.
- Consider using data visualization tools or software to create charts and graphs to better understand sales trends over time.

<u>b. External Factors Data</u>:

- For marketing campaigns, collect data on the timing, content, and channels used for each campaign.
- For holidays, create a calendar of relevant holidays and special events that could affect sales.
- For economic indicators, access data from government sources or reputable financial data providers.

Data Preprocessing

- Describe the significance of data cleaning and preparation.
- Highlight the tasks like handling missing values and converting categorical features into numerical ones.

```python
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Load the dataset (replace 'data.csv' with your dataset file)
data = pd.read_csv('data.csv')

# Split the data into features (X) and target (y)
X = data.drop('electricity_price', axis=1)
y = data['electricity_price']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Standardize the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Feature Engineering

- Explain the role of creating additional features to capture patterns.
- Emphasize capturing seasonal trends, overall trends, and external influences.

```
# You can create additional features based on your domain knowledge
# For example, adding lag features for time-series data, or engineering seasonal patterns

# Example: Adding lag features
for lag in range(1, 6):  # Create lag features for 1 to 5 time steps
data[f'lag_{lag}'] = data['electricity_price'].shift(lag)

# Remove rows with missing values due to the lag feature creation
data = data.dropna()
```

Model Selection

- Discuss the importance of choosing the right forecasting model.
- List suitable regression algorithms (e.g., Linear Regression, Random Forest, XGBoost).

```
# Import machine learning models
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor

# Initialize the models
model_lr = LinearRegression()
model_rf = RandomForestRegressor()
model_xgb = XGBRegressor()

# You can further tune hyperparameters for each model
```

Model Training

- Explain the process of training the chosen forecasting model.
- Mention that it uses the preprocessed data.

```
# Fit the models to the training data
model_lr.fit(X_train, y_train)
model_rf.fit(X_train, y_train)
model_xgb.fit(X_train, y_train)
```

Evaluation Metrics

- Introduce evaluation metrics to assess model performance.
- Explain Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

```
# Import evaluation metrics
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Make predictions on the test set
y_pred_lr = model_lr.predict(X_test)
y_pred_rf = model_rf.predict(X_test)
y_pred_xgb = model_xgb.predict(X_test)

# Evaluate the models
mae_lr = mean_absolute_error(y_test, y_pred_lr)
mae_rf = mean_absolute_error(y_test, y_pred_rf)
mae_xgb = mean_absolute_error(y_test, y_pred_xgb)

# You can also calculate RMSE, R-squared, and other relevant metrics
```

Model Performance

- Display model evaluation results visually.
- Discuss how these results reflect the model's accuracy in predicting demand.

Conclusion:

- Summarize the key steps in the demand forecasting workflow.
- Highlight the importance of accurate forecasting for decision-making.

THANKING YOU