FUTURE SALES PREDICTION

IBM NAAN MUDHALVAN

PROJECT SUBMISSION PHASE-4

In the analysis of the "Future Sales Prediction" dataset, we conducted a comprehensive series of data analysis steps to create an accurate prediction model. The process began with Exploratory Data Analysis (EDA) to understand the dataset's characteristics. Subsequently, we performed data preprocessing, including outlier detection and handling using the winzoring technique, as well as data normalization using the min-max method. We then developed multiple models, including Linear Regression, Ridge Regression, Lasso Regression, Decision Tree, and Random Forest, all of which were evaluated through cross-validation. The model evaluation results revealed that Random Forest outperformed others, yielding an average Mean Squared Error (MSE) of 10.32%, Root Mean Squared Error (RMSE) of 8.09%, Mean Absolute Error (MAE) of 5.99%, and an R-squared value of 94.27%. Additionally, we conducted classic assumption tests, including tests for linearity, homoscedasticity, normality, multicollinearity, outliers, and independence, to ensure the validity of our model. These results provide in-depth insights into the quality of our prediction model and its relevance in the context of future sales forecasting.

Dataset is taken from kaggle competition and it can be downloaded from here:

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

Feature Engineering:

Feature engineering includes remodeling raw data into a format that successfully represents the underlying patterns within the data. It involves selecting, combining, and crafting attributes that capture the relationships between variables, enhancing the predictive power of machine learning models. These engineered features acts as the input for algorithms, using progressed performance and robustness.

Feature Tools:

A python library centered on automated feature engineering, particularly for time-series and relational data. It automates producing new features by leveraging domain-specific knowledge and entity relationships.

Applications: Creating time-based features, aggregating data over different time intervals, and handling a couple of associated data tables.

Features explanation:

- **TV:** This feature represents the amount of advertising budget spent on television media for a product or service in a certain period, for example in thousands of dollars (USD).
- **Radio:** This feature represents the amount of advertising budget spent on radio media in the same period as TV.
- **Newspaper:** This feature represents the amount of advertising budget spent in newspapers or print media in the same period as TV and Radio.
- **Sales:** This feature represents product or service sales data in the same period as advertising expenditure on TV, Radio and Newspaper.

Load Data

Input:

1. import pandas as pd

   df = pd.read_csv('/kaggle/input/future-sales-prediction/Sales.csv')

   df.head()

2. import pandas as pd

   df = pd.read_csv('/kaggle/input/future-sales-prediction/Sales.csv')

   df.shape

3. import pandas as pd

   df = pd.read_csv('/kaggle/input/future-sales-prediction/Sales.csv')

   df.info()

4. import pandas as pd

   df = pd.read_csv('/kaggle/input/future-sales-prediction/Sales.csv')

   df.describe().T

Exploratory Data Analysis (EDA)

5. import plotly.express as px

figure = px.scatter(df, x='Sales', y='TV', size='TV', trendline='ols', title='Relationship Between Sales and TV Advertising')

figure.update_traces(marker=dict(line=dict(width=2, color='DarkSlateGrey')), selector=dict(mode='markers'))

figure.update_layout(

xaxis_title='Sales',

yaxis_title='TV Advertising',

legend_title='TV Ad Size',

plot_bgcolor='white'

)

figure.show()

6. figure= px.scatter(df, x='Sales', y='Newspaper', size='Newspaper', trendline='ols', title='Relationship Between Sales and Newspaper Advertising')

figure.update_traces(marker=dict(line=dict(width=2, color='DarkSlateGrey')), selector=dict(mode='markers'))

figure.update_layout(

xaxis_title='Sales',

yaxis_title='Newspaper Advertising',

legend_title='Newspaper Ad Size',

plot_bgcolor='white'

)

```
figure.show()
```

7. px.scatter(df, x='Sales', y='Radio', size='Radio', trendline='ols', title='Relationship Between Sales and Radio Advertising')

figure.update_traces(marker=dict(line=dict(width=2, color='DarkSlateGrey')), selector=dict(mode='markers'))

figure.update_layout(

    xaxis_title='Sales',

    yaxis_title='Radio Advertising',

    legend_title='Radio Ad Size',

    plot_bgcolor='white'

)

figure.show()