import numpy as np import pandas as pd import datetime import seaborn as sns import matplotlib.pyplot as plt path1 ='.\Marketing Data[1].csv' path2 ='.\Revenue Data 1[1].csv' path3 ='.\Revenue Data 2[1].csv' path4 ='.\Visitors Data[1].xlsx' df1=pd.read_csv(path1) df2=pd.read_csv(path2) df3=pd.read_csv(path3) df4=pd.read excel(path4) df1.info() In [4]: <class 'pandas.core.frame.DataFrame'> RangeIndex: 185 entries, 0 to 184 Data columns (total 3 columns): Column Non-Null Count Dtype -----0 Date 185 non-null object 1 Marketing Spend 178 non-null float64 182 non-null dtypes: float64(1), object(2) memory usage: 4.5+ KB df1 **Promo Date Marketing Spend 0** 09/11/2020 651.375 No Promo 1298.250 **Promotion Red 1** 10/11/2020 **2** 11/11/2020 1559.375 Promotion Blue **3** 12/11/2020 1801.750 No Promo No Promo **4** 13/11/2020 2614.500 **180** 08/05/2021 1121.875 No Promo **181** 09/05/2021 871.000 No Promo **182** 10/05/2021 NaN NaN **183** 11/05/2021 NaN NaN 184 12/05/2021 NaN NaN 185 rows × 3 columns df1['Date']=pd.to datetime(df1['Date'],format='%d/%m/%Y') dfl.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 185 entries, 0 to 184 Data columns (total 3 columns): Non-Null Count Dtype # Column 185 non-null datetime64[ns] 0 Date Marketing Spend 178 non-null float64
Promo 182 non-null object 1 dtypes: datetime64[ns](1), float64(1), object(1) memory usage: 4.5+ KB df2.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 133 entries, 0 to 132 Data columns (total 7 columns): Column Non-Null Count Dtype 133 non-null object 129 non-null float64 0 Date Week ID 129 non-null float64 2 Month Number 129 non-null Month_ID 129 non-null 3 float64 4 133 non-null Year int64 5 Day_Name 129 non-null object 125 non-null Revenue float64 dtypes: float64(4), int64(1), object(2) memory usage: 7.4+ KB df2['Date']=pd.to datetime(df2['Date'],format='%d/%m/%Y') In [9]: df3.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 56 entries, 0 to 55 Data columns (total 7 columns): Column Non-Null Count Dtype _____ Date 56 non-null
Week_ID 53 non-null
Month Number 53 non-null
Month_ID 53 non-null 0 object Date float64 1 float64 Month_ID 53 non ...
53 non-null 3 float64 float64 4 5 Day_Name 53 non-null object 52 non-null Revenue float64 dtypes: float64(5), object(2) memory usage: 3.2+ KB df3['Date']=pd.to datetime(df3['Date'],format='%d/%m/%Y') df4.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 185 entries, 0 to 184 Data columns (total 2 columns): # Column Non-Null Count Dtype datetime64[ns] float64 0 Date 185 non-null 1 Visitors 177 non-null dtypes: datetime64[ns](1), float64(1) memory usage: 3.0 KB df rev=pd.concat([df2,df3]) df rev Week_ID Month Number Month_ID Year Day_Name Revenue Date 0 2020-11-09 34.0 11.0 11.0 2020.0 Monday 465.0 **1** 2020-11-10 34.0 11.0 11.0 2020.0 Tuesday 10386.0 2 2020-11-11 34.0 11.0 11.0 2020.0 Wednesday 12475.0 **3** 2020-11-12 34.0 11.0 11.0 2020.0 Thursday 11712.0 **4** 2020-11-13 11.0 2020.0 Friday 10000.0 34.0 11.0 60.0 17.0 2021.0 51 2021-05-08 5.0 Saturday 5927.0 **52** 2021-05-09 61.0 5.0 17.0 2021.0 Sunday 3861.0 **53** 2021-05-10 NaN NaN NaN NaN NaN NaN 2021-05-11 54 NaN NaN NaN NaN NaN NaN **55** 2021-05-12 NaN NaN NaN NaN NaN NaN 189 rows × 7 columns In [14]: df m r = df1.merge(df rev,on='Date',how='left') df_m_r Out[14]: Marketing Month Day_Name Revenue **Date** Week_ID Month_ID Promo Year Spend Number 2020-651.375 0 No Promo 34.0 11.0 11.0 2020.0 Monday 465.0 11-09 2020-Promotion 1298.250 1 34.0 11.0 11.0 2020.0 Tuesday 10386.0 11-10 Red 2020-Promotion 2 1559.375 34.0 11.0 11.0 2020.0 Wednesday 12475.0 11-11 Blue 2020-1801.750 3 No Promo 34.0 11.0 11.0 2020.0 Thursday 11712.0 11-12 2020-2614.500 No Promo 4 34.0 11.0 11.0 2020.0 Friday 10000.0 11-13 2021-184 1121.875 No Promo 60.0 5.0 17.0 2021.0 Saturday 5927.0 05-08 2021-185 871.000 No Promo 61.0 5.0 17.0 2021.0 Sunday 3861.0 05-09 2021-186 NaN NaN NaN NaN NaN NaN NaN NaN 05-10 2021-187 NaN NaN NaN NaN NaN NaN NaN NaN 05-11 2021-188 NaN NaN NaN NaN NaN NaN NaN NaN 05-12 189 rows × 9 columns df m r Marketing Month **Date** Promo Week_ID Month_ID Day_Name Spend Number 2020-0 2020.0 651.375 No Promo 34.0 11.0 11.0 Monday 465.0 11-09 2020-Promotion 1298.250 34.0 11.0 2020.0 10386.0 11.0 Tuesday 11-10 Red 2020-Promotion 2 1559.375 34.0 11.0 11.0 2020.0 Wednesday 12475.0 11-11 Blue 2020-3 1801.750 34.0 2020.0 11712.0 No Promo 11.0 11.0 Thursday 11-12 2020-4 2020.0 2614.500 No Promo 34.0 11.0 11.0 Friday 10000.0 11-13 2021-184 1121.875 No Promo 60.0 5.0 17.0 2021.0 Saturday 5927.0 05-08 2021-185 871.000 61.0 17.0 2021.0 No Promo 5.0 Sunday 3861.0 05-09 2021-186 NaN NaN NaN NaN NaN NaN NaN NaN 05-10 2021-187 NaN NaN NaN NaN NaN NaN NaN NaN 05-11 2021-188 NaN NaN NaN NaN NaN NaN NaN NaN 05-12 189 rows × 9 columns df = df m r.merge(df4,on='Date',how='left') df Marketing Month **Date** Promo Week_ID Month_ID Year Day_Name Revenue Visitors Spend Number 2020-651.375 No Promo 34.0 11.0 11.0 2020.0 707.0 Monday 465.0 11-09 2020-Promotion 1298.250 11.0 2020.0 34.0 11.0 Tuesday 10386.0 1455.0 11-10 Red 2020-Promotion 1559.375 34.0 11.0 11.0 2020.0 Wednesday 12475.0 1520.0 11-11 Blue 2020-1801.750 11.0 2020.0 No Promo 34.0 11.0 Thursday 11712.0 1726.0 11-12 2020-2614.500 No Promo 34.0 11.0 2020.0 10000.0 11.0 Friday 2134.0 11-13 2021-1121.875 No Promo 60.0 5.0 17.0 2021.0 Saturday 5927.0 1483.0 05-08 2021-185 871.000 17.0 2021.0 No Promo 61.0 5.0 Sunday 3861.0 1303.0 05-09 2021-186 NaN NaN NaN NaN NaN NaN NaN NaN NaN 05-10 2021-187 NaN NaN NaN NaN NaN NaN NaN NaN NaN 05-11 2021-188 NaN NaN NaN NaN NaN NaN NaN NaN NaN 05-12 189 rows × 10 columns df.info() <class 'pandas.core.frame.DataFrame'> Int64Index: 189 entries, 0 to 188 Data columns (total 10 columns): Non-Null Count Dtype Column 0 189 non-null datetime64[ns] Marketing Spend 182 non-null float64 1 2 186 non-null object Promo float64 182 non-null 3 Week ID Month Number 182 non-null float64 5 182 non-null float64 Month ID float64 6 186 non-null Year 7 Day Name 182 non-null object 178 non-null Visitors 181 non-null float64 dtypes: datetime64[ns](1), float64(7), object(2) memory usage: 16.2+ KB In [18]: df.dropna(inplace=True) df.info() In [19]: <class 'pandas.core.frame.DataFrame'> Int64Index: 169 entries, 0 to 185 Data columns (total 10 columns): Non-Null Count Dtype Column Date 169 non-null 0 datetime64[ns] 1 Marketing Spend 169 non-null float64 2 Promo 169 non-null object 3 Week ID 169 non-null float64 4 Month Number 169 non-null float64 5 Month_ID 169 non-null float64 6 Year 169 non-null float64 7 Day_Name 169 non-null object 8 Revenue 169 non-null float64 169 non-null Visitors float64 dtypes: datetime64[ns](1), float64(7), object(2) memory usage: 14.5+ KB sns.pairplot(df) <seaborn.axisgrid.PairGrid at 0x230db73f550> 3000 P 2500 2000 1500 1000 60 55 0 50 ⊆ ¥ 45 40 35 12 10 17 16 15 £ 14 13 12 11 2021.0 2020.8 2020.6 2020.4 25000 20000 15000 10000 4000 3000 7000 1000 1000 4000 000 2000 30 Marketing Spend 3000 2020.5 Year 2021.0 20000 sns.heatmap(df.corr(),annot = True) <AxesSubplot:> - 1.00 -0.061 0.058 -0.041 -0.059 0.74 Marketing Spend -0.84 0.75 -0.62 0.99 0.8 -0.015 -0.03 Week_ID · 0.50 0.058 -0.62 -0.59 -0.96 0.028 0.016 Month Number 0.25 0.99 0.78 -0.0073 -0.014 -0.041 -0.59 1 Month ID 0.00 0.059 -0.96 0.78 1 -0.024 -0.017 -0.25Year -0.50 -0.015 0.028 -0.0073 -0.024 0.74 Revenue -0.75 0.84 -0.03 0.016 -0.014 -0.017 1 Visitors pend enne, umber Visitors \Box ₽ léar 듚 Marketing S Revenue and Visitors have positive correlation with Marketing Spend, sns.catplot(x='Day_Name',y='Revenue',data = df,kind = 'bar') <seaborn.axisgrid.FacetGrid at 0x230dce70e50> 16000 14000 12000 10000 Revenue 8000 6000 4000 2000 0 Monday TuesdaWednesdaThursday Friday Saturday Sunday Day_Name Thursday and Friday are the most revenue days sns.catplot(x='Promo', y='Revenue', data = df, kind = 'box') <seaborn.axisgrid.FacetGrid at 0x230de679220> 25000 20000 Revenue 15000 10000 5000 0 No Promo Promotion Red Promotion Blue Promo Promotion blue is the best promotion sns.scatterplot(x='Marketing Spend',y='Revenue',data = df,hue ='Promo') <AxesSubplot:xlabel='Marketing Spend', ylabel='Revenue'> Promo No Promo 25000 Promotion Red Promotion Blue 20000 Revenue 15000 10000 5000 0 500 1000 1500 2000 2500 3000 3500 Marketing Spend Revenue is directly proportional to Marketing Spend even for higher values of marketing spend in case of Promotion blue. whereas revenue for Promotion red starts to flatten after Marketing spend of £2000. df.columns Out[26]: Index(['Date', 'Marketing Spend', 'Promo', 'Week_ID', 'Month Number', 'Month_ID', 'Year', 'Day_Name', 'Revenue', 'Visitors'], dtype='object') features=['Marketing Spend','Visitors', 'Day_Name','Promo'] X=df[features] y=df['Revenue'] from sklearn.model_selection import train_test_split X_train, X_valid, y_train, y_valid = train_test_split(X, y,train_size=0.8,test_size=0 from sklearn.preprocessing import OrdinalEncoder ordinal encoder = OrdinalEncoder() label X train = X train.copy() label_X_valid = X_valid.copy() label_X_train[['Promo','Day_Name']] = ordinal_encoder.fit_transform(X_train[['Promo', label_X_valid[['Promo','Day_Name']] = ordinal_encoder.transform(X_valid[['Promo','Day_ from sklearn.linear_model import LinearRegression from sklearn.metrics import mean_absolute_error def mae_lm(X_train, X_valid, y_train, y_valid): model = LinearRegression() model.fit(X_train, y_train) preds = model.predict(X_valid) return mean_absolute_error(y_valid, preds) print('Mean absolute error for Linear Regression model is {}'.format(mae_lm(label_X_t) Mean absolute error for Linear Regression model is 2519.85161049745 from sklearn.ensemble import RandomForestRegressor def mae_rf(X_train, X_valid, y_train, y_valid): model = RandomForestRegressor(n_estimators=200,random_state=0) model.fit(X_train, y_train) preds = model.predict(X_valid) return mean_absolute_error(y_valid, preds) print('Mean absolute error for RandomForest model is {}'.format(mae_rf(label_X_train, Mean absolute error for RandomForest model is 1590.0152941176473 X_test = pd.DataFrame({'Marketing Spend':[5000],'Visitors':[8000],'Day_Name':['Friday In [34]: label_X_test = X_test.copy() label_X_test[['Promo','Day_Name']] = ordinal_encoder.transform(X_test[['Promo','Day_Name']) model = RandomForestRegressor(n_estimators=200,random_state=0) model.fit(label_X_train, y_train) preds = model.predict(label_X_valid) print('Mean absolute error for RandomForest model is {}'.format(mean_absolute_error Mean absolute error for RandomForest model is 1590.0152941176473 In [38]: prediction = model.predict(label_X_test) print('Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotic Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotion Red o n Friday is [14045.83] X_test = pd.DataFrame({'Marketing Spend':[5000],'Visitors':[8000],'Day_Name':['Friday In [41]: label_X_test = X_test.copy() label_X_test[['Promo','Day_Name']] = ordinal_encoder.transform(X_test[['Promo','Day_Name']]) prediction = model.predict(label_X_test) print('Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotic Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotion Blue on Friday is [25037.07]