Task for Exploratory Analysis and Regression Model

Introduction:

In the web application a simple task of exploratory analysis and regression modeling is done.For creating a user interface for the task , streamlit has been used. Streamlit is a framework to build machine learning and data science web applications. Streamlit can integrate with python libraries like Numpy, Pandas etc.
The First feature is to upload the data file (excel file) , and then the dataframe and shape is displayed . On click graphs such as time series plot and correlation plot are displayed . Then the test and train data sets along with the time series graph of predicted values is displayed.

Libraries Used:

1.Pandas - It is a Python package that offers various data structures and operations for manipulating numerical data and time series. Using pandas we read the input file and convert it into data frames and perform operations on them.

2.Matplotlib - It is a library for creating graphs

3.Seaborn- Seaborn is a data visualization library in Python, as an extension to Matplotlib.Random distributions can be visualized and correlations can be found.

4.sklearn -It is a library for machine learning in python. It helps in classification , regression, clustering and more.

Code:

1.Importing Libraries

```
import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
```

2.Used file_uploader in streamlit to upload the file

```
st.title('Task for Exploratory analysis and Regression Model')
st.text('In this web page a user can upload file , get the graphs and predictions by clicking on buttons')

# To upload file
file_upload = st.file_uploader('Upload an excel file')

# Checking if the file is uploaded and not empty
if file_upload is not None:
    #Reading the file using pandas into a dataframe
    df = pd.read_excel(file_upload)

    st.write(df.head())
    st.write(df.shape)

    #checking for null values
    df.isnull().sum()
```

3. Used read_excel in pandas to read the csv file

4. After reading the file checked for null values , as there are no null values and all the columns are correlated didn't perform data cleaning.

5. To plot time series graph , clubbed the YY , MM , DD columns into date column and HH , mm into time column and merged them together and plotted them against output.

```
    # Merging Date , Time coloumns
    cols = ["YY", "MM", "DD"]
    df1['Date'] = df1[cols].apply(lambda x: '-'.join(x.values.astype(str)), axis="columns")
    df1['Date'] = pd.to_datetime(df1['Date'])

    cols = ["HH", "mm"]
    df1['Time'] = df1[cols].apply(lambda x: ':'.join(x.values.astype(str)), axis="columns")
    df1['Time'] = pd.to_datetime(df1['Time'])

    cols = ['Date','Time']
    df1['DT'] = df1[cols].apply(lambda x:' '.join(x.values.astype(str)), axis="columns")


if st.button('display time series graph'):
    st.subheader('Plot of Data')
    fig, ax = plt.subplots(1,1)
    ax.scatter(x=df1['DT'], y=df1['Output'])
    ax.set_xlabel('Date Time')
    ax.set_ylabel('Output')
    fig.set_figwidth(10)
    st.pyplot(fig)
```

6.For the correlation graph , used the corr() function .

```
if st.button('display correlation plot'):
    st.subheader('Correlation graph')
    fig, ax = plt.subplots()
    sns.heatmap(df1.corr(), ax=ax)
    fig.set_figwidth(5)
    st.write(fig)
```

7.Manually splitting the data as we need the values in sequence as the dataset in 8:2 ratio. Splitting the train and test data set and displaying them on the web page.

```
if st.button('Split data into train and test'):
    #manully splitting data in 8:2 ratio
    val = int(0.8 * df.shape[0])

    #slicing the dataset
    train = df[:val]
    test = df[val:]

    X_train = train[train.columns[~df.columns.isin(['Output'])]]
    Y_train = train[["Output"]]
    X_test = test[test.columns[~df.columns.isin(['Output'])]]
    Y_test = test[["Output"]]
    st.caption("Training Data Set")
    st.write(train)
    st.caption("Testing Data Set")
    st.write(test)
```

8.Using LinearRegression().fit() of the sklearn library to perform the linear regression and fit the data.

9.Used .predict() function to predict the the values on the test data set(x_test)

10.Plotted values for predicted values against date and time in the test data set(x_test)

```
#fitting the data
model = LinearRegression().fit(X_train, Y_train)
Y_Pred = model.predict(X_test)

#plot for the predicted values
st.header('Plot of Predicted Values')
fig1, ax1 = plt.subplots(1,1)
cols = ["YY", "MM", "DD"]
X_test['Date'] = X_test[cols].apply(lambda x: '-'.join(x.values.astype(str)), axis="columns")
X_test['Date'] = pd.to_datetime(X_test['Date'])
cols = ["HH", "mm"]
X_test['Time'] = X_test[cols].apply(lambda x: ':'.join(x.values.astype(str)), axis="columns")
X_test['Time'] = pd.to_datetime(X_test['Time'])

cols = ['Date', 'Time']
X_test['DateandTime'] = X_test[cols].apply(lambda x:' '.join(x.values.astype(str)), axis="columns")

ax1.scatter(x=X_test['DateandTime'], y=Y_Pred)
ax1.set_xlabel('Date and Time')
ax1.set_ylabel('Predicted values')
fig1.set_figwidth(10)
st.pyplot(fig1)
```

Used the st.button in streamlit to display the buttons on the webpage.