Media Company Analysis

In [398…
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Data Preprocessing

In [399…
```
data = pd.read_csv('mediacompany.csv')
data.head()
```

Out[399]:

	Date	Views_show	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A	Unnamed: 7
0	3/1/2017	183738	1260228	1706478	1060860448	0	0	NaN
1	3/2/2017	193763	1270561	1690727	1031846645	0	0	NaN
2	3/3/2017	210479	1248183	1726157	1010867575	0	0	NaN
3	3/4/2017	240061	1492913	1855353	1079194579	1	0	NaN
4	3/5/2017	446314	1594712	2041418	1357736987	0	0	NaN

In [400…
```
del data['Unnamed: 7']
data.head()
```

Out[400]:

	Date	Views_show	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A
0	3/1/2017	183738	1260228	1706478	1060860448	0	0
1	3/2/2017	193763	1270561	1690727	1031846645	0	0
2	3/3/2017	210479	1248183	1726157	1010867575	0	0
3	3/4/2017	240061	1492913	1855353	1079194579	1	0
4	3/5/2017	446314	1594712	2041418	1357736987	0	0

In [401…
```
import datetime
import calendar
```

In [402…
```
data.isnull().sum()
```

Out[402]:
```
Date                0
Views_show          0
Visitors            0
Views_platform      0
Ad_impression       0
Cricket_match_india 0
Character_A         0
dtype: int64
```

Adding Day column to dataset

In [403…
```
def findDay(date):
    born = datetime.datetime.strptime(date, '%d %m %Y').weekday()
    return (calendar.day_name[born])
```

In [404…
```
date = data.loc[: ,'Date'].astype('datetime64').dt.strftime('%d %m %Y')

for i in date.head():
    print(findDay(i),":", i)
```

```
Wednesday : 01 03 2017
Thursday : 02 03 2017
Friday : 03 03 2017
Saturday : 04 03 2017
Sunday : 05 03 2017
```

In [405…
```
day = np.array([findDay(x) for x in date])
data['Day'] = day
data.head()
```

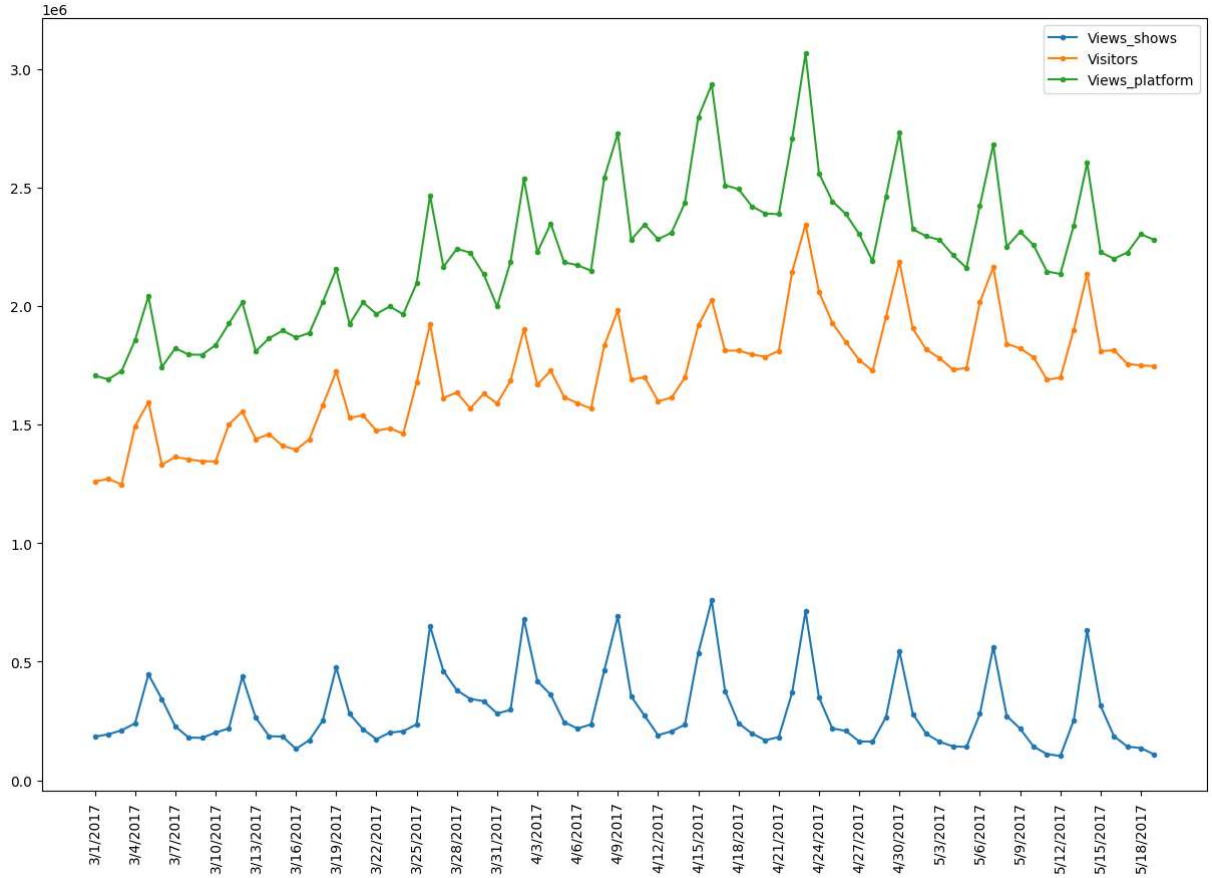
Out[405]:

	Date	Views_show	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A	Day
0	3/1/2017	183738	1260228	1706478	1060860448	0	0	Wednesday
1	3/2/2017	193763	1270561	1690727	1031846645	0	0	Thursday
2	3/3/2017	210479	1248183	1726157	1010867575	0	0	Friday
3	3/4/2017	240061	1492913	1855353	1079194579	1	0	Saturday
4	3/5/2017	446314	1594712	2041418	1357736987	0	0	Sunday

In [406…
```
orig_data = data.copy(deep=True)
```

Data Analysis

In [407…
```
plt.figure(figsize=(15, 10))
plt.plot(data.Date, data.Views_show, label='Views_shows', marker= '.')
plt.plot(data.Date, data.Visitors, label='Visitors', marker= '.')
plt.plot(data.Date, data.Views_platform, label='Views_platform', marker= '.')
plt.xticks(data.Date[::3], rotation=90)
plt.legend()
plt.show()
```



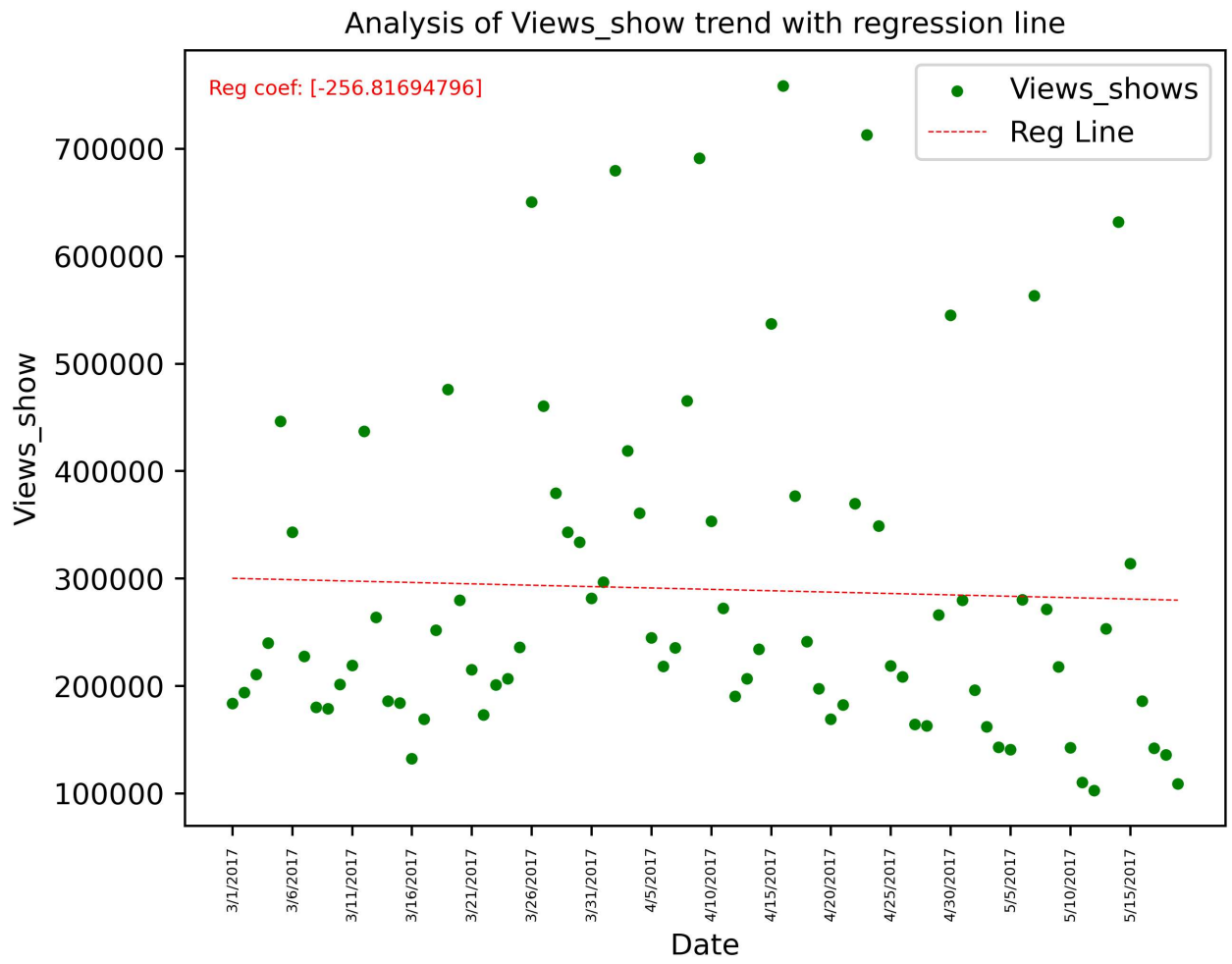
Views_show Analysis with regression line

In [408…
```
from sklearn.linear_model import LinearRegression
```

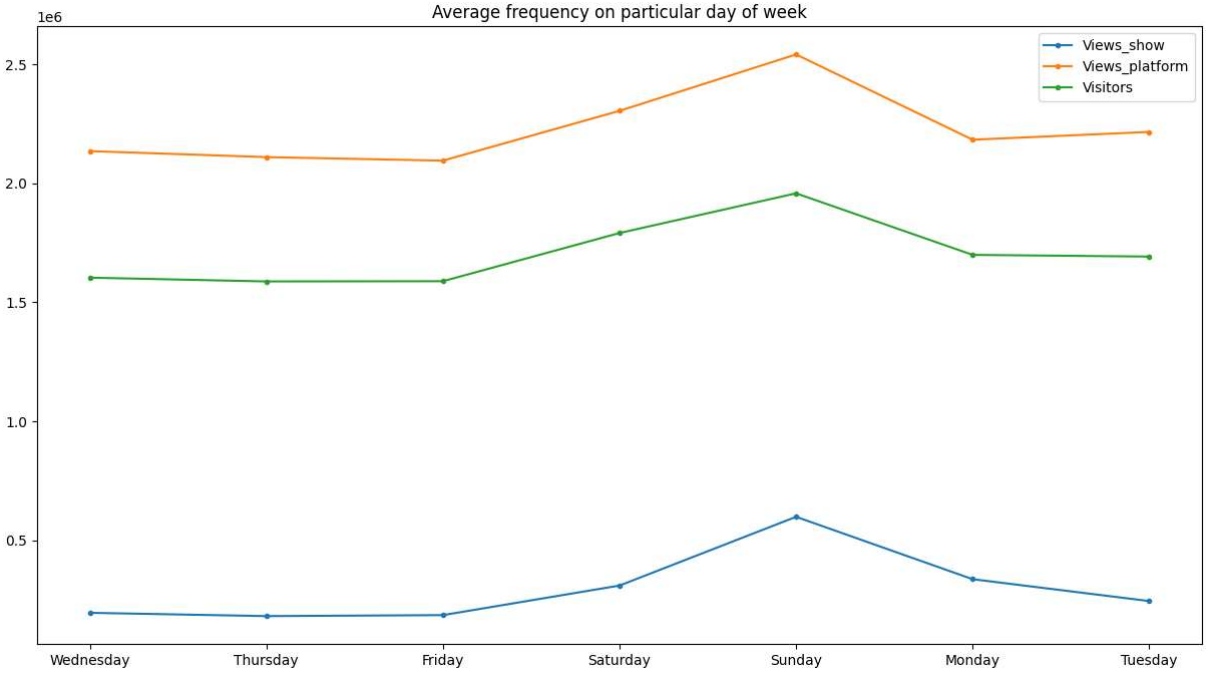
In [409…
```
Views_model = LinearRegression()
Views_model.fit(np.arange(data.Date.shape[0]).reshape(data.Date.shape[0], 1), data.Views_show)
Views_pred = Views_model.predict(np.arange(data.Date.shape[0]).reshape(data.Date.shape[0], 1))
Views_pred
```

Out[409]:
```
array([300046.89444444, 299790.07749648, 299533.26054852, 299276.44360056,
       299019.6266526 , 298762.80970464, 298505.99275668, 298249.17580872,
       297992.35886076, 297735.5419128 , 297478.72496484, 297221.90801688,
       296965.09106892, 296708.27412096, 296451.457173  , 296194.64022504,
       295937.82327707, 295681.00632911, 295424.18938115, 295167.37243319,
       294910.55548523, 294653.73853727, 294396.92158931, 294140.10464135,
       293883.28769339, 293626.47074543, 293369.65379747, 293112.83684951,
       292856.01990155, 292599.20295359, 292342.38600563, 292085.56905767,
       291828.7521097 , 291571.93516174, 291315.11821378, 291058.30126582,
       290801.48431786, 290544.6673699 , 290287.85042194, 290031.03347398,
       289774.21652602, 289517.39957806, 289260.5826301 , 289003.76568214,
       288746.94873418, 288490.13178622, 288233.31483826, 287976.4978903 ,
       287719.68094233, 287462.86399437, 287206.04704641, 286949.23009845,
       286692.41315049, 286435.59620253, 286178.77925457, 285921.96230661,
       285665.14535865, 285408.32841069, 285151.51146273, 284894.69451477,
       284637.87756681, 284381.06061885, 284124.24367089, 283867.42672293,
       283610.60977496, 283353.792827  , 283096.97587904, 282840.15893108,
       282583.34198312, 282326.52503516, 282069.7080872 , 281812.89113924,
       281556.07419128, 281299.25724332, 281042.44029536, 280785.6233474 ,
       280528.80639944, 280271.98945148, 280015.17250352, 279758.35555556])
```

In [410…
```
plt.figure(dpi = 441)
plt.scatter(data.Date, data.Views_show, label='Views_shows', marker= '.', color='green')
plt.plot(np.arange(data.Date.shape[0]).reshape(data.Date.shape[0], 1), Views_pred, linewidth= 0.5, color='red', linestyle='--', label = "Reg Line")
plt.text(-2, 750000, f'Reg coef: {Views_model.coef_}', fontsize=7, color='red')
plt.xticks(data.Date[::5], rotation=90, fontsize=5)
plt.xlabel("Date")
plt.ylabel("Views_show")
plt.legend()
plt.title('Analysis of Views_show trend with regression line', fontsize=10)
plt.show()
```



In [411…
```
data.groupby(orig_data.Day, sort=False).mean(numeric_only=True)[['Views_show', 'Views_platform', 'Visitors']].astype("int").plot(kind='line',marker='.' , figsize=(15, 8))
plt.xticks(rotation=0)
plt.title('Average frequency on particular day of week')
plt.xlabel('')
plt.show()
```



Normalising data for further analysis

In [412…
```
from sklearn.preprocessing import MinMaxScaler
```

In [413…
```
cols_to_scale = ['Views_show', 'Visitors', 'Views_platform', "Ad_impression"]
scaler = MinMaxScaler()
data[cols_to_scale] = scaler.fit_transform(data[cols_to_scale])
```

In [414…
```
data.head()
```

Out[414]:

	Date	Views_show	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A	Day
0	3/1/2017	0.123787	0.010985	0.011466	0.035372	0	0	Wednesday
1	3/2/2017	0.139066	0.020409	0.000000	0.014844	0	0	Thursday
2	3/3/2017	0.164544	0.000000	0.025792	0.000000	0	0	Friday
3	3/4/2017	0.209631	0.223199	0.119842	0.048345	1	0	Saturday
4	3/5/2017	0.523988	0.316041	0.255290	0.245426	0	0	Sunday

Adding bias to the normalised data

In [415…

```
data.Visitors = data.Visitors + 1
data.Views_platform = data.Views_platform + 2
data.Ad_impression = data.Ad_impression + 3
data.head()
```

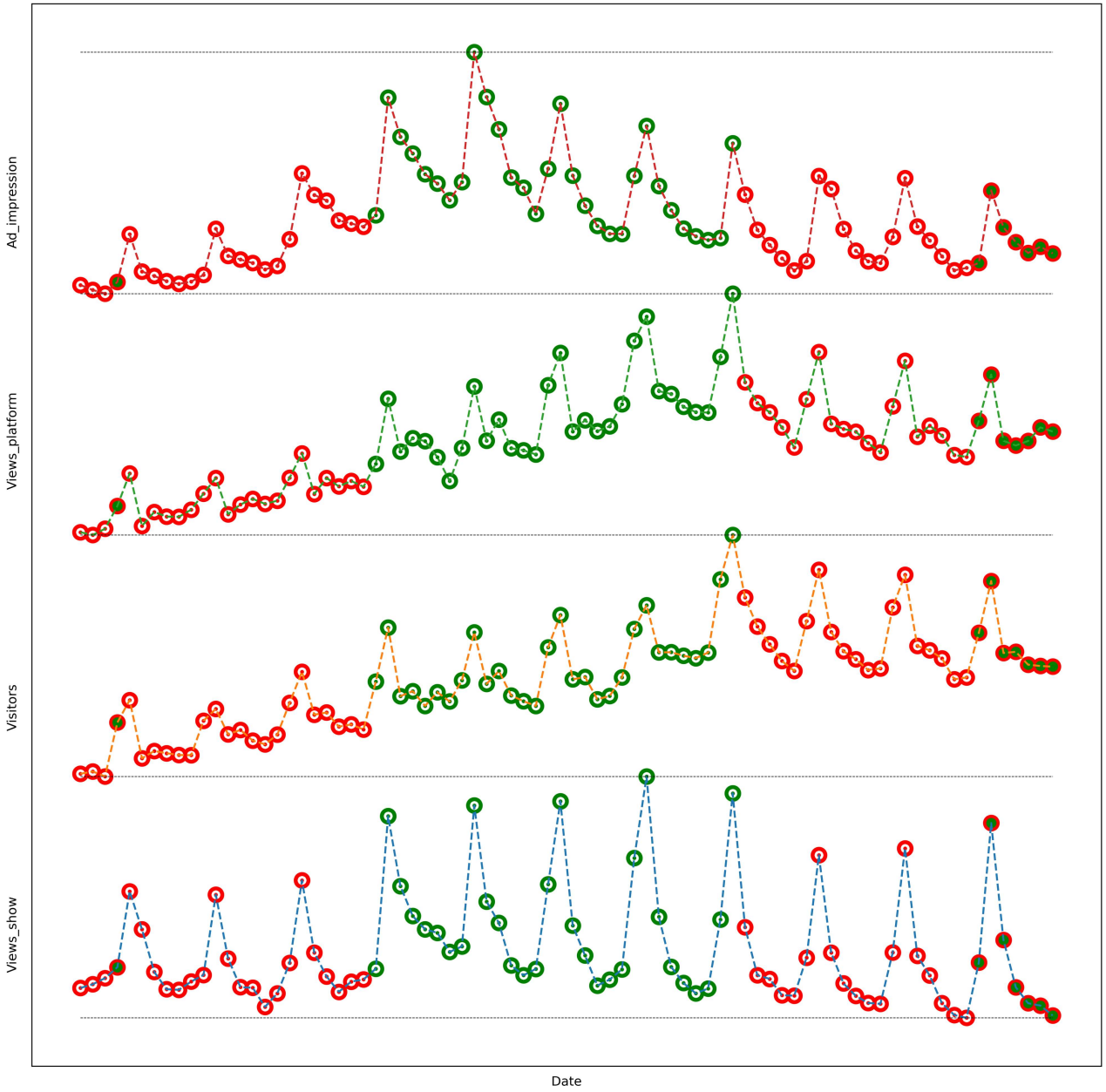
Out[415]:

	Date	Views_show	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A	Day
0	3/1/2017	0.123787	1.010985	2.011466	3.035372	0	0	Wednesday
1	3/2/2017	0.139066	1.020409	2.000000	3.014844	0	0	Thursday
2	3/3/2017	0.164544	1.000000	2.025792	3.000000	0	0	Friday
3	3/4/2017	0.209631	1.223199	2.119842	3.048345	1	0	Saturday
4	3/5/2017	0.523988	1.316041	2.255290	3.245426	0	0	Sunday

Complete Data Visualization in Single plot

In [416…

```
plt.figure(figsize=(15, 15), dpi=441)

# Views_show
plt.plot(data.Date, data.Views_show, label= 'Views', linestyle='--')
plt.scatter(data.groupby('Character_A').get_group(0).Date, data.groupby('Character_A').get_group(0).Views_show, label="Character Not Changed",  color='red', linewidth=10, marker
plt.scatter(data.groupby('Character_A').get_group(1).Date, data.groupby('Character_A').get_group(1).Views_show, label="Character Changed", color='green', linewidth=10, marker='.
plt.scatter(data.groupby('Cricket_match_india').get_group(1).Date, data.groupby('Cricket_match_india').get_group(1).Views_show, label='Cricket', color='green', linewidth=2)

# Visitors
plt.plot(data.Date, data.Visitors, label = 'Visitors', linestyle='--')
plt.scatter(data.groupby('Character_A').get_group(0).Date, data.groupby('Character_A').get_group(0).Visitors, label="Character Not Changed",  color='red', linewidth=10, marker='
plt.scatter(data.groupby('Character_A').get_group(1).Date, data.groupby('Character_A').get_group(1).Visitors, label="Character Changed", color='green', linewidth=10, marker='.')
plt.scatter(data.groupby('Cricket_match_india').get_group(1).Date, data.groupby('Cricket_match_india').get_group(1).Visitors, label='Cricket', color='green', linewidth=2)

# Views_platform
plt.plot(data.Date, data.Views_platform, label = 'Views_platfrom', linestyle='--')
plt.scatter(data.groupby('Character_A').get_group(0).Date, data.groupby('Character_A').get_group(0).Views_platform, label="Character Not Changed",  color='red', linewidth=10, mar
plt.scatter(data.groupby('Character_A').get_group(1).Date, data.groupby('Character_A').get_group(1).Views_platform, label="Character Changed", color='green', linewidth=10, marke
plt.scatter(data.groupby('Cricket_match_india').get_group(1).Date, data.groupby('Cricket_match_india').get_group(1).Views_platform, label='Cricket', color='green', linewidth=2)

# Ad_impression
plt.plot(data.Date, data.Ad_impression, label = "Ad_impression", linestyle='--')
plt.scatter(data.groupby('Character_A').get_group(0).Date, data.groupby('Character_A').get_group(0).Ad_impression, label="Character Not Changed",  color='red', linewidth=10, marl
plt.scatter(data.groupby('Character_A').get_group(1).Date, data.groupby('Character_A').get_group(1).Ad_impression, label="Character Changed", color='green', linewidth=10, marker
plt.scatter(data.groupby('Cricket_match_india').get_group(1).Date, data.groupby('Cricket_match_india').get_group(1).Ad_impression, label='Cricket', color='green', linewidth=2)

# Line seperator
plt.plot(data.Date, [0]*data.Date.shape[0],  label = "Line Seperator" , color='black', linestyle= '--', linewidth=0.5)
plt.plot(data.Date, [1]*data.Date.shape[0],  label = "Line Seperator" , color='black', linestyle= '--', linewidth=0.5)
plt.plot(data.Date, [2]*data.Date.shape[0],  label = "Line Seperator" , color='black', linestyle= '--', linewidth=0.5)
plt.plot(data.Date, [3]*data.Date.shape[0],  label = "Line Seperator" , color='black', linestyle= '--', linewidth=0.5)
plt.plot(data.Date, [4]*data.Date.shape[0],  label = "Line Seperator" , color='black', linestyle= '--', linewidth=0.5)


# plt.ylabel('Views                                    Visitors                                    Views_platform                                    Ad_impression')
plt.text(-6, 0.2, "Views_show", rotation=90)
plt.text(-6, 1.2, "Visitors", rotation=90)
plt.text(-6, 2.2, "Views_platform", rotation=90)
plt.text(-6, 3.2, "Ad_impression", rotation=90)

plt.xlabel('Date')
plt.xticks([])
plt.yticks([])
plt.show()
```



In [417…

```
views_char_no_change = int(np.average(orig_data.groupby('Character_A').get_group(0).Views_show))
print(f'Avg Views when character not changed: {views_char_no_change}')
views_char_no_change = int(np.average(orig_data.groupby('Character_A').get_group(0).Views_show))
print(f'Avg Views when character changed: {views_char_change}')

print(f'Diffence: {views_char_change - views_char_no_change}')
```

```
Avg Views when character not changed: 241899
Avg Views when character changed: 369907
Diffence: 128008
```

Complete Data Visualization in Single plot on every day of week

In [418…

```
def View_trend(df, day):

    dd = df.groupby(df.Day)

    # Views_show
    plt.plot(dd.get_group(day).Date, dd.get_group(day).Views_show, label = "Views_show")
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(0).Date, dd.get_group(day).groupby('Character_A').get_group(0).Views_show, label = "Character Not Changed", mar
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(1).Date, dd.get_group(day).groupby('Character_A').get_group(1).Views_show, label = "Character Changed", marker=
#     plt.scatter(dd.get_group(day).groupby('Cricket_match_india').get_group(0).Date, dd.get_group(day).groupby('Cricket_match_india').get_group(0).Views_show, label='Not Cricket
    plt.scatter(dd.get_group(day).groupby('Cricket_match_india').get_group(1).Date, dd.get_group(day).groupby('Cricket_match_india').get_group(1).Views_show, label='Cricket', co


    # Visitors
    plt.plot(dd.get_group(day).Date, dd.get_group(day).Visitors, label = "Visitors", marker='.')
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(0).Date, dd.get_group(day).groupby('Character_A').get_group(0).Visitors, label = "Character Not Changed", marke
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(1).Date, dd.get_group(day).groupby('Character_A').get_group(1).Visitors, label = "Character Changed", marker='

    # Views_platform
    plt.plot(dd.get_group(day).Date, dd.get_group(day).Views_platform, label = "Views_platform", marker='.')
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(0).Date, dd.get_group(day).groupby('Character_A').get_group(0).Views_platform, label = "Character Not Changed"
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(1).Date, dd.get_group(day).groupby('Character_A').get_group(1).Views_platform, label = "Character Changed", mar

    # Ad_impression
    plt.plot(dd.get_group(day).Date, dd.get_group(day).Ad_impression, label='Ad_impression', marker='.')
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(0).Date, dd.get_group(day).groupby('Character_A').get_group(0).Ad_impression, label = "Character Not Changed",
    plt.scatter(dd.get_group(day).groupby('Character_A').get_group(1).Date, dd.get_group(day).groupby('Character_A').get_group(1).Ad_impression, label = "Character Changed", mar

    # Line seperator
    plt.plot(dd.get_group(day).Date, [0]*dd.get_group(day).Date.shape[0], color='black', linestyle= '--', linewidth=0.5)
    plt.plot(dd.get_group(day).Date, [1]*dd.get_group(day).Date.shape[0], color='black', linestyle= '--', linewidth=0.5)
    plt.plot(dd.get_group(day).Date, [2]*dd.get_group(day).Date.shape[0], color='black', linestyle= '--', linewidth=0.5)
    plt.plot(dd.get_group(day).Date, [3]*dd.get_group(day).Date.shape[0], color='black', linestyle= '--', linewidth=0.5)
```

```
    plt.plot(dd.get_group(day).Date, [4]*dd.get_group(day).Date.shape[0], color='black', linestyle= '--', linewidth=0.5)

    plt.xticks([])
    plt.yticks([])
    plt.title(day)
    plt.xlabel('Date')
    plt.text(-1.1, 0.2, "Views_show", rotation=90)
    plt.text(-1.1, 1.2, "Visitors", rotation=90)
    plt.text(-1.1, 2.2, "Views_platform", rotation=90)
    plt.text(-1.1, 3.2, "Ad_impression", rotation=90)
```

In [419…

```
plt.figure(figsize=(25, 25), dpi=441)
plt.suptitle("Complete Data Visualization in Single plot on every day of week", fontsize=30)

# plt.subplot(341)
# View_trend(data, 'Monday')

# plt.subplot(342)
# View_trend(data, 'Tuesday')

# plt.subplot(343)
# View_trend(data, 'Wednesday')

# plt.subplot(344)
# View_trend(data, 'Thursday')

# plt.subplot(345)
# View_trend(data, 'Friday')

# plt.subplot(346)
# View_trend(data, 'Saturday')

# plt.subplot(347)
# View_trend(data, 'Sunday')


count = 1
for i in ['Sunday', 'Monday', 'Tuesday', 'Wednesday', "Thursday", 'Friday', "Saturday"]:
    coord = 340 + count
    plt.subplot(coord)
    View_trend(data, i)
    count+=1


plt.subplot(348)
plt.xticks([])
plt.yticks([])
plt.plot([0], [0], label = "Views_show")
plt.plot([0], [0], label = "Visitors")
plt.plot([0], [0], label = "Views_platform")
plt.plot([0], [0], label = "Ad_impression")


plt.scatter([0], [0], label = "Character Not Changed", marker='.', color='red', linewidth=10)
plt.scatter([0], [0], label = "Character Changed", marker='.', color='green', linewidth=10)
#     plt.scatter(dd.get_group(day).groupby('Cricket_match_india').get_group(0).Date, dd.get_group(day).groupby('Cricket_match_india').get_group(0).Views_show, label='Not Cricket
plt.scatter([0], [0], label='Cricket', color='green', linewidth=2)
plt.axis('off')
plt.legend(loc='center', fontsize=20)
plt.show()
```

Multiple Linear Regression

Data Preparation

In [420…]
```
data.head()
```

Out[420]:

	Date	Views_show	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A	Day
0	3/1/2017	0.123787	1.010985	2.011466	3.035372	0	0	Wednesday
1	3/2/2017	0.139066	1.020409	2.000000	3.014844	0	0	Thursday
2	3/3/2017	0.164544	1.000000	2.025792	3.000000	0	0	Friday
3	3/4/2017	0.209631	1.223199	2.119842	3.048345	1	0	Saturday
4	3/5/2017	0.523988	1.316041	2.255290	3.245426	0	0	Sunday

Checking correlation in data

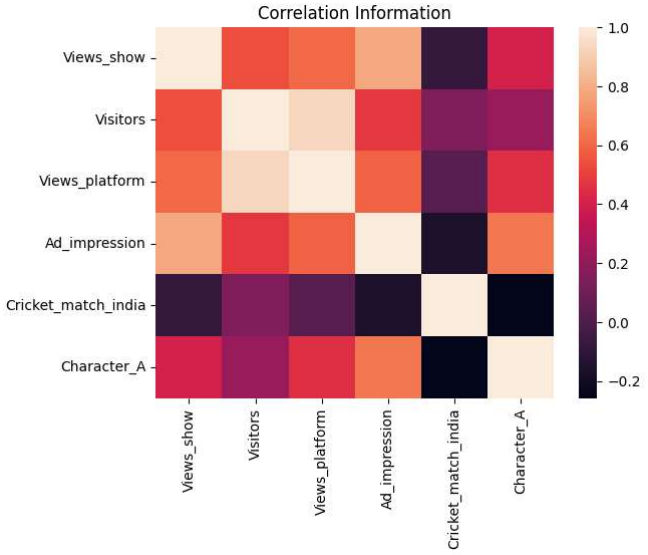
In [421…]
```
heatMap = data[['Views_show', 'Visitors', 'Views_platform', 'Ad_impression', 'Cricket_match_india', 'Character_A']].corr()
heatMap
```

Out[421]:

	Views_show	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A
Views_show	1.000000	0.535290	0.604279	0.785673	-0.083154	0.402533
Visitors	0.535290	1.000000	0.935832	0.478598	0.147210	0.223093
Views_platform	0.604279	0.935832	1.000000	0.587003	0.036575	0.452375
Ad_impression	0.785673	0.478598	0.587003	1.000000	-0.163936	0.644354
Cricket_match_india	-0.083154	0.147210	0.036575	-0.163936	1.000000	-0.258199
Character_A	0.402533	0.223093	0.452375	0.644354	-0.258199	1.000000

In [422…]
```
import seaborn as sns
```

In [423…]
```
sns.heatmap(heatMap)
plt.title("Correlation Information")
plt.show()
```



Spliting into Independent variable and Dependent variable

In [427…]
```
# Independent Variable
x = data[['Visitors', 'Views_platform', 'Ad_impression', 'Cricket_match_india', 'Character_A', 'Day']]
x.head()
```

Out[427]:

	Visitors	Views_platform	Ad_impression	Cricket_match_india	Character_A	Day
0	1.010985	2.011466	3.035372	0	0	Wednesday
1	1.020409	2.000000	3.014844	0	0	Thursday
2	1.000000	2.025792	3.000000	0	0	Friday
3	1.223199	2.119842	3.048345	1	0	Saturday
4	1.316041	2.255290	3.245426	0	0	Sunday

In [428…]
```
# Dependent Variable
y = data['Views_show']
y.head()
```

Out[428]:
```
0    0.123787
1    0.139066
2    0.164544
3    0.209631
4    0.523988
Name: Views_show, dtype: float64
```

Encoding Day into nominal order

In [430…]
```
ohe = OneHotEncoder(drop= 'first', sparse_output = False, dtype=np.int32)
day_encode_x = ohe.fit_transform(x[['Day']])
```

In [431…]
```
x_new = np.hstack((x[['Visitors', 'Views_platform', 'Ad_impression', 'Cricket_match_india', 'Character_A']].values, day_encode_x))
pd.DataFrame(x_new).head()
```

Out[431]:

	0	1	2	3	4	5	6	7	8	9	10
0	1.010985	2.011466	3.035372	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0
1	1.020409	2.000000	3.014844	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
2	1.000000	2.025792	3.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	1.223199	2.119842	3.048345	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
4	1.316041	2.255290	3.245426	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0

Spliting into train and test data

In [432…
```
from sklearn.model_selection import train_test_split
```

In [433…
```
x_train, x_test, y_train, y_test = train_test_split(x_new, y, test_size=0.30, random_state=43)
```

In [434…
```
print(f'x_train shape: {x_train.shape}')
print(f'x_test shape: {x_test.shape}')
print(f'y_train shape: {y_train.shape}')
print(f'y_test shape: {y_test.shape}')
```

```
x_train shape: (56, 11)
x_test shape: (24, 11)
y_train shape: (56,)
y_test shape: (24,)
```

Model Building

In [435…
```
from sklearn.linear_model import LinearRegression
```

In [436…
```
model = LinearRegression()
model.fit(x_train, y_train)
```

Out[436]:
```
▾ LinearRegression
LinearRegression()
```

Model Analysis

In [437…
```
pred_train = model.predict(x_train)
```

In [438…
```
pred_test = model.predict(x_test)
```

In [439…
```
pred_all = model.predict(x_new)
```

Coefficient of Determination

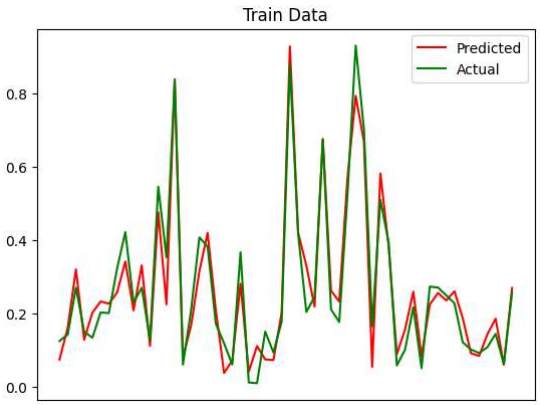
In [440…
```
from sklearn.metrics import r2_score
```

In [441…
```
print(f"r2 value of training: {r2_score(y_train, pred_train)}")
print(f"r2 value of testing: {r2_score(y_test, pred_test)}")
print(f"r2 value of complete Data: {r2_score(y, pred_all)}")
```

```
r2 value of training: 0.9299500404621269
r2 value of testing: 0.8848391732344474
r2 value of complete Data: 0.9124629862012006
```

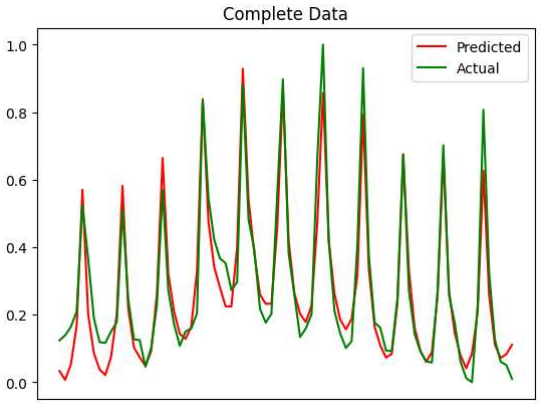
Visual Presentation

In [442…
```
plt.plot(np.arange(pred_train.shape[0]), pred_train, color='red', label="Predicted")
plt.plot(np.arange(y_train.shape[0]), y_train, color='green', label="Actual")
plt.title("Train Data")
plt.legend()
plt.xticks([])
plt.show()
```



In [443…
```
plt.plot(np.arange(pred_test.shape[0]), pred_test, color='red',  label='Predicted')
plt.plot(np.arange(pred_test.shape[0]), y_test, color='green', label='Actual')
plt.xticks([])
plt.title("Test Data")
plt.legend()
plt.show()
```

In [444…
```
plt.plot(np.arange(pred_all.shape[0]), pred_all, color='red',  label='Predicted')
plt.plot(np.arange(pred_all.shape[0]), y, color='green', label='Actual')
plt.xticks([])
plt.title("Complete Data")
plt.legend()
plt.show()
```



Regression Model using OLS (Ordinal Least Square)

In [445…
```
import statsmodels.api as sm
```

In [446…
```
x_with_const_train = sm.add_constant(x_train)
x_with_const_test = sm.add_constant(x_test)
x_with_const_all = sm.add_constant(x_new)
```

In [447…
```
ols_model_train = sm.OLS(y_train, x_with_const_train)
ols_model_test = sm.OLS(y_test, x_with_const_test)
ols_model_all = sm.OLS(y, x_with_const_all)
```

In [448…
```
result_train = ols_model_train.fit()
result_test = ols_model_test.fit()
result_all = ols_model_all.fit()
```

In [449…
```
result_train.params
```

Out[449]:
```
const   -1.395784
x1      -0.201554
x2       0.241020
x3       0.387666
x4      -0.012705
x5       0.035199
x6       0.124749
x7       0.129748
x8       0.429276
x9      -0.042407
x10      0.008466
x11     -0.028849
dtype: float64
```

In [450…
```
print(result_train.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             Views_show   R-squared:                       0.930
Model:                            OLS   Adj. R-squared:                  0.912
Method:                 Least Squares   F-statistic:                     53.10
Date:                Wed, 16 Nov 2022   Prob (F-statistic):           1.03e-21
Time:                        22:37:28   Log-Likelihood:                 81.800
No. Observations:                  56   AIC:                            -139.6
Df Residuals:                      44   BIC:                            -115.3
Df Model:                          11
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.3958      0.374     -3.732      0.001      -2.150      -0.642
x1            -0.2016      0.197     -1.025      0.311      -0.598       0.195
x2             0.2410      0.214      1.126      0.266      -0.190       0.672
x3             0.3877      0.085      4.562      0.000       0.216       0.559
x4            -0.0127      0.028     -0.454      0.652      -0.069       0.044
x5             0.0352      0.034      1.025      0.311      -0.034       0.104
x6             0.1247      0.036      3.458      0.001       0.052       0.197
x7             0.1297      0.034      3.867      0.000       0.062       0.197
x8             0.4293      0.048      9.008      0.000       0.333       0.525
x9            -0.0424      0.031     -1.362      0.180      -0.105       0.020
x10            0.0085      0.038      0.224      0.824      -0.068       0.085
x11           -0.0288      0.030     -0.958      0.344      -0.090       0.032
==============================================================================
Omnibus:                        2.084   Durbin-Watson:                   1.912
Prob(Omnibus):                  0.353   Jarque-Bera (JB):                1.876
Skew:                           0.440   Prob(JB):                        0.391
Kurtosis:                       2.833   Cond. No.                         232.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [451…
```
print(result_test.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             Views_show   R-squared:                       0.988
Model:                            OLS   Adj. R-squared:                  0.976
Method:                 Least Squares   F-statistic:                     87.27
Date:                Wed, 16 Nov 2022   Prob (F-statistic):           1.07e-09
Time:                        22:37:29   Log-Likelihood:                 49.664
No. Observations:                  24   AIC:                            -75.33
Df Residuals:                      12   BIC:                            -61.19
Df Model:                          11
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -2.2386      0.448     -4.992      0.000      -3.216      -1.262
x1            -1.4804      0.206     -7.198      0.000      -1.928      -1.032
x2             1.5473      0.195      7.924      0.000       1.122       1.973
x3             0.2338      0.122      1.913      0.080      -0.032       0.500
x4             0.2529      0.061      4.119      0.001       0.119       0.387
x5            -0.1391      0.054     -2.575      0.024      -0.257      -0.021
x6             0.3196      0.044      7.281      0.000       0.224       0.415
x7             0.2924      0.047      6.184      0.000       0.189       0.395
x8             0.5476      0.052     10.450      0.000       0.433       0.662
x9             0.0090      0.042      0.216      0.832      -0.081       0.099
x10            0.1104      0.037      2.957      0.012       0.029       0.192
x11            0.0425      0.040      1.055      0.312      -0.045       0.130
==============================================================================
Omnibus:                        2.684   Durbin-Watson:                   1.648
Prob(Omnibus):                  0.261   Jarque-Bera (JB):                1.211
Skew:                           0.228   Prob(JB):                        0.546
Kurtosis:                       4.001   Cond. No.                         257.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [452…
```
print(result_all.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             Views_show   R-squared:                       0.927
Model:                            OLS   Adj. R-squared:                  0.915
Method:                 Least Squares   F-statistic:                     78.04
Date:                Wed, 16 Nov 2022   Prob (F-statistic):           4.09e-34
Time:                        22:37:30   Log-Likelihood:                 106.94
No. Observations:                  80   AIC:                            -189.9
Df Residuals:                      68   BIC:                            -161.3
Df Model:                          11
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.5755      0.281     -5.599      0.000      -2.137      -1.014
x1            -0.5916      0.148     -4.004      0.000      -0.886      -0.297
x2             0.6595      0.156      4.221      0.000       0.348       0.971
x3             0.3030      0.075      4.063      0.000       0.154       0.452
x4             0.0046      0.028      0.165      0.869      -0.051       0.060
x5             0.0081      0.030      0.266      0.791      -0.053       0.069
x6             0.1781      0.033      5.346      0.000       0.112       0.245
x7             0.1711      0.030      5.650      0.000       0.111       0.231
x8             0.4928      0.040     12.183      0.000       0.412       0.574
x9            -0.0207      0.028     -0.734      0.465      -0.077       0.036
x10            0.0439      0.031      1.439      0.155      -0.017       0.105
x11           -0.0122      0.029     -0.426      0.671      -0.069       0.045
==============================================================================
Omnibus:                        2.785   Durbin-Watson:                   1.090
Prob(Omnibus):                  0.248   Jarque-Bera (JB):                2.760
Skew:                           0.428   Prob(JB):                        0.252
Kurtosis:                       2.694   Cond. No.                         187.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Model building using Artificial Neural Network

Using keras Library based on Tensorflow

In [385…

```
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense
from keras import optimizers
from keras.optimizers import SGD
```

In [386…

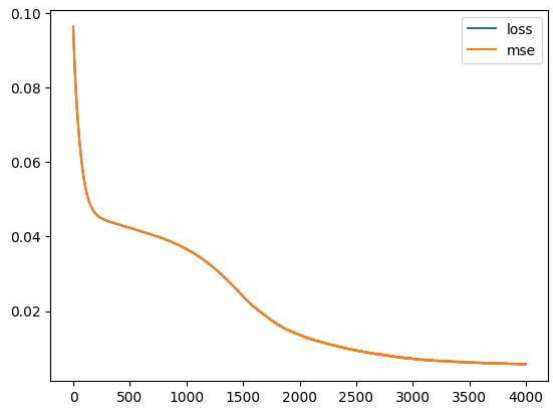
```
model_ann = Sequential()
model_ann.add(Dense(11, input_dim=11, activation='relu'))
model_ann.add(Dense(11, activation='relu'))
model_ann.add(Dense(11, activation='relu'))
model_ann.add(Dense(11, activation='relu'))
model_ann.add(Dense(11, activation='relu'))
model_ann.add(Dense(11, activation='relu'))
model_ann.add(Dense(11, activation='relu'))

model_ann.add(Dense(1, activation='sigmoid'))
model_ann.compile(loss='mean_squared_error', optimizer='sgd', metrics=['mse'])
history = model_ann.fit(x_train, y_train, epochs=4000)
```

```
Epoch 1/4000
2/2 [==============================] - 1s 15ms/step - loss: 0.0964 - mse: 0.0964
Epoch 2/4000
2/2 [==============================] - 0s 1ms/step - loss: 0.0953 - mse: 0.0953
Epoch 3/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0943 - mse: 0.0943
Epoch 4/4000
2/2 [==============================] - 0s 11ms/step - loss: 0.0934 - mse: 0.0934
Epoch 5/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0925 - mse: 0.0925
Epoch 6/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0917 - mse: 0.0917
Epoch 7/4000
2/2 [==============================] - 0s 15ms/step - loss: 0.0908 - mse: 0.0908
Epoch 8/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0900 - mse: 0.0900
Epoch 9/4000
2/2 [==============================] - 0s 996us/step - loss: 0.0892 - mse: 0.0892
Epoch 10/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0885 - mse: 0.0885
Epoch 11/4000
2/2 [==============================] - 0s 12ms/step - loss: 0.0877 - mse: 0.0877
Epoch 12/4000
2/2 [==============================] - 0s 1ms/step - loss: 0.0870 - mse: 0.0870
Epoch 13/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0863 - mse: 0.0863
Epoch 14/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0856 - mse: 0.0856
Epoch 15/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0849 - mse: 0.0849
Epoch 16/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0843 - mse: 0.0843
Epoch 17/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0836 - mse: 0.0836
Epoch 18/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0830 - mse: 0.0830
Epoch 19/4000
2/2 [==============================] - 0s 17ms/step - loss: 0.0824 - mse: 0.0824
Epoch 20/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0818 - mse: 0.0818
Epoch 21/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0812 - mse: 0.0812
Epoch 22/4000
2/2 [==============================] - 0s 16ms/step - loss: 0.0806 - mse: 0.0806
Epoch 23/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0800 - mse: 0.0800
Epoch 24/4000
2/2 [==============================] - 0s 667us/step - loss: 0.0795 - mse: 0.0795
Epoch 25/4000
2/2 [==============================] - 0s 16ms/step - loss: 0.0789 - mse: 0.0789
Epoch 26/4000
2/2 [==============================] - 0s 14ms/step - loss: 0.0784 - mse: 0.0784
Epoch 27/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0779 - mse: 0.0779
Epoch 28/4000
2/2 [==============================] - 0s 2ms/step - loss: 0.0773 - mse: 0.0773
Epoch 29/4000
2/2 [==============================] - 0s 14ms/step - loss: 0.0768 - mse: 0.0768
Epoch 30/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0763 - mse: 0.0763
Epoch 31/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0759 - mse: 0.0759
Epoch 32/4000
2/2 [==============================] - 0s 13ms/step - loss: 0.0754 - mse: 0.0754
Epoch 33/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0749 - mse: 0.0749
Epoch 34/4000
2/2 [==============================] - 0s 2ms/step - loss: 0.0745 - mse: 0.0745
Epoch 35/4000
2/2 [==============================] - 0s 13ms/step - loss: 0.0740 - mse: 0.0740
Epoch 36/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0736 - mse: 0.0736
Epoch 37/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0731 - mse: 0.0731
Epoch 38/4000
2/2 [==============================] - 0s 12ms/step - loss: 0.0727 - mse: 0.0727
Epoch 39/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0722 - mse: 0.0722
Epoch 40/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0718 - mse: 0.0718
Epoch 41/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0714 - mse: 0.0714
Epoch 42/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0709 - mse: 0.0709
Epoch 43/4000
2/2 [==============================] - 0s 16ms/step - loss: 0.0705 - mse: 0.0705
Epoch 44/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0701 - mse: 0.0701
Epoch 45/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0697 - mse: 0.0697
Epoch 46/4000
2/2 [==============================] - 0s 15ms/step - loss: 0.0693 - mse: 0.0693
Epoch 47/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0689 - mse: 0.0689
Epoch 48/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0685 - mse: 0.0685
Epoch 49/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0682 - mse: 0.0682
Epoch 50/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0678 - mse: 0.0678
Epoch 51/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0674 - mse: 0.0674
Epoch 52/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0671 - mse: 0.0671
Epoch 53/4000
2/2 [==============================] - 0s 12ms/step - loss: 0.0667 - mse: 0.0667
Epoch 54/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0664 - mse: 0.0664
Epoch 55/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0660 - mse: 0.0660
Epoch 56/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0657 - mse: 0.0657
Epoch 57/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0653 - mse: 0.0653
Epoch 58/4000
2/2 [==============================] - 0s 15ms/step - loss: 0.0650 - mse: 0.0650
Epoch 59/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0647 - mse: 0.0647
Epoch 60/4000
2/2 [==============================] - 0s 1ms/step - loss: 0.0644 - mse: 0.0644
```

```
Epoch 3961/4000
2/2 [==============================] - 0s 2ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3962/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3963/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0058 - mse: 0.0058
Epoch 3964/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3965/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0058 - mse: 0.0058
Epoch 3966/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3967/4000
2/2 [==============================] - 0s 2ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3968/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3969/4000
2/2 [==============================] - 0s 7ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3970/4000
2/2 [==============================] - 0s 2ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3971/4000
2/2 [==============================] - 0s 10ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3972/4000
2/2 [==============================] - 0s 3ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3973/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3974/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3975/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3976/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0058 - mse: 0.0058
Epoch 3977/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3978/4000
2/2 [==============================] - 0s 11ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3979/4000
2/2 [==============================] - 0s 3ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3980/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3981/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0057 - mse: 0.0057
Epoch 3982/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3983/4000
2/2 [==============================] - 0s 7ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3984/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0058 - mse: 0.0058
Epoch 3985/4000
2/2 [==============================] - 0s 9ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3986/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3987/4000
2/2 [==============================] - 0s 3ms/step - loss: 0.0058 - mse: 0.0058
Epoch 3988/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0057 - mse: 0.0057
Epoch 3989/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3990/4000
2/2 [==============================] - 0s 6ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3991/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3992/4000
2/2 [==============================] - 0s 6ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3993/4000
2/2 [==============================] - 0s 993us/step - loss: 0.0057 - mse: 0.0057
Epoch 3994/4000
2/2 [==============================] - 0s 13ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3995/4000
2/2 [==============================] - 0s 0s/step - loss: 0.0057 - mse: 0.0057
Epoch 3996/4000
2/2 [==============================] - 0s 7ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3997/4000
2/2 [==============================] - 0s 2ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3998/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0057 - mse: 0.0057
Epoch 3999/4000
2/2 [==============================] - 0s 5ms/step - loss: 0.0057 - mse: 0.0057
Epoch 4000/4000
2/2 [==============================] - 0s 4ms/step - loss: 0.0057 - mse: 0.0057
```

In [387…

```
pd.DataFrame(history.history).plot()
plt.show()
```



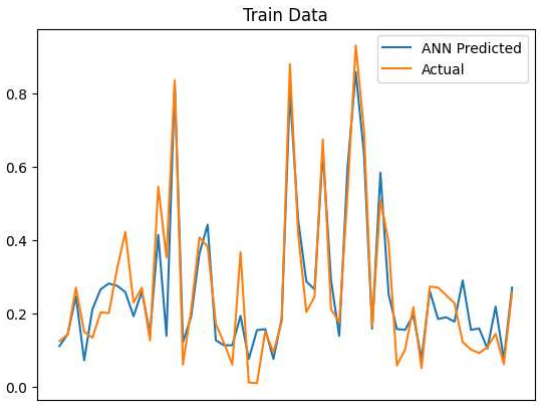
In [388…

```
pred_train_ann = model_ann.predict(x_train)
```

```
2/2 [==============================] - 0s 0s/step
```

In [389…

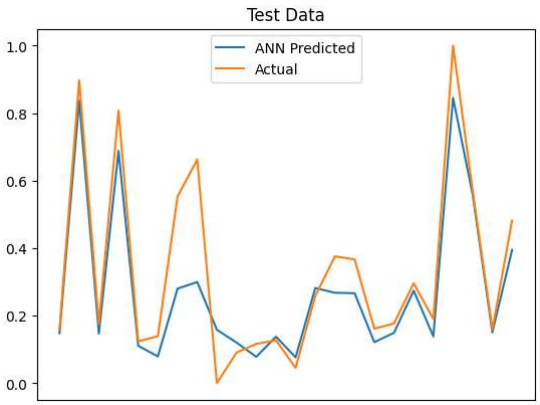
```
plt.plot(np.arange(56), pred_train_ann, label="ANN Predicted")
plt.plot(np.arange(56), y_train, label='Actual')
plt.legend()
plt.title("Train Data")
plt.xticks([])
plt.show()
```



In [391…
```
pred_test_ann = model_ann.predict(x_test)
```

```
1/1 [==============================] - 0s 24ms/step
```

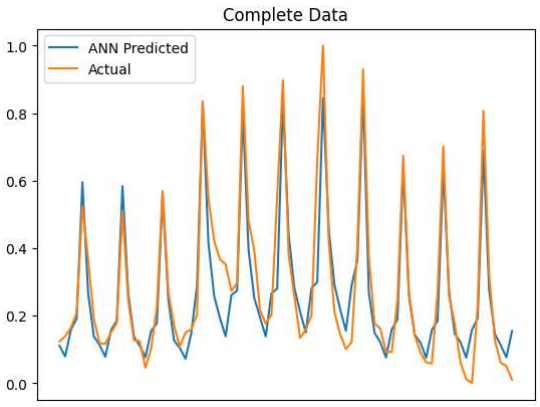
In [392…
```
plt.plot(np.arange(24), pred_test_ann, label="ANN Predicted")
plt.plot(np.arange(24), y_test, label='Actual')
plt.legend(loc='upper center')
plt.title("Test Data")
plt.xticks([])
plt.show()
```



In [393…
```
pred_all_ann = model_ann.predict(x_new)
```

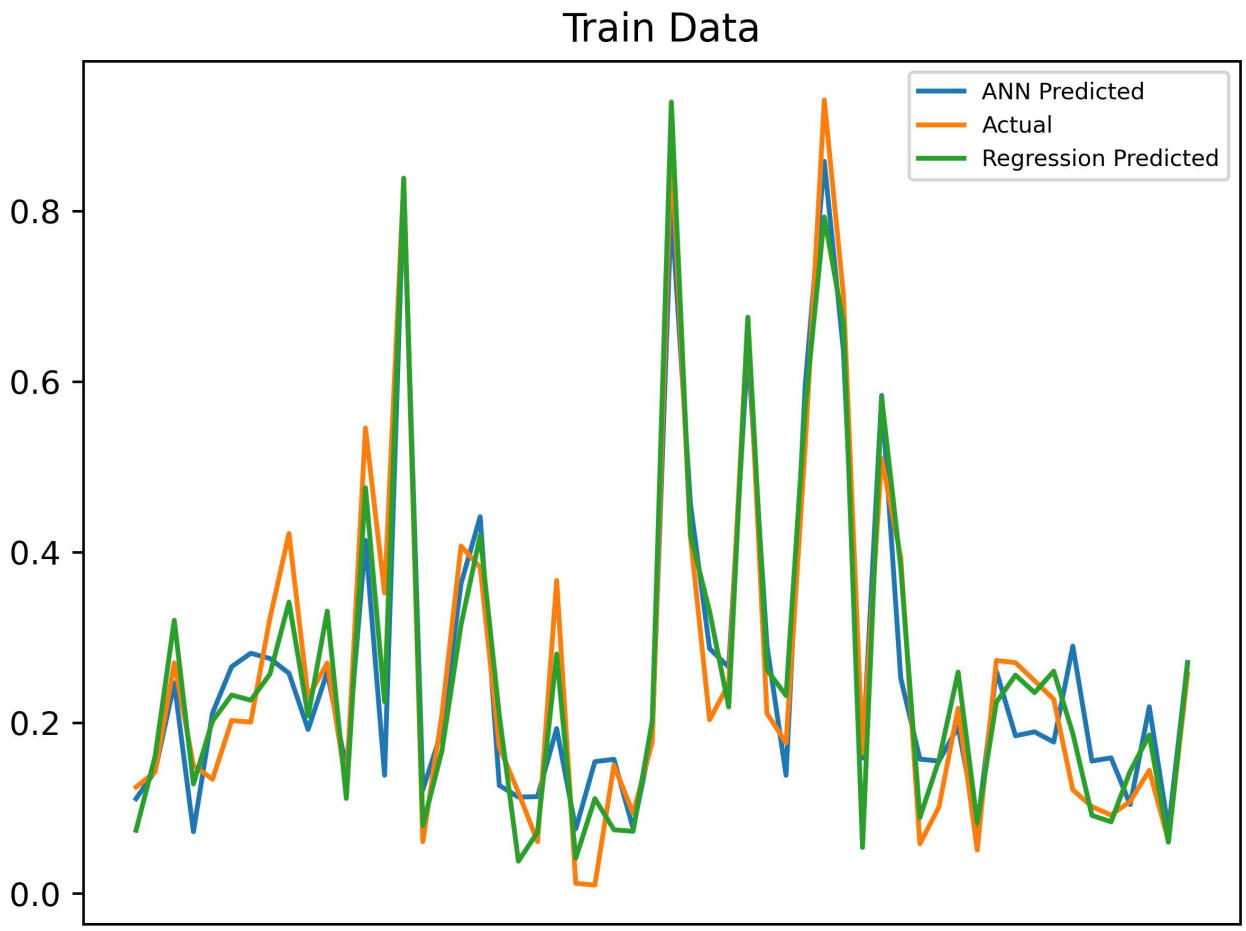
```
3/3 [==============================] - 0s 2ms/step
```

In [397…
```
plt.plot(np.arange(80), pred_all_ann, label="ANN Predicted")
plt.plot(np.arange(80), y, label='Actual')
plt.legend(loc='upper left')
plt.title("Complete Data")
plt.xticks([])
plt.show()
```



ANN Model vs Regression Model

In [467…
```
plt.figure(dpi=441)
plt.plot(np.arange(56), pred_train_ann, label="ANN Predicted")
plt.plot(np.arange(56), y_train, label='Actual')
plt.plot(np.arange(56), pred_train, label = "Regression Predicted")
plt.legend(loc='upper right', fontsize= 7)
plt.title("Train Data")
plt.xticks([])
plt.show()
```

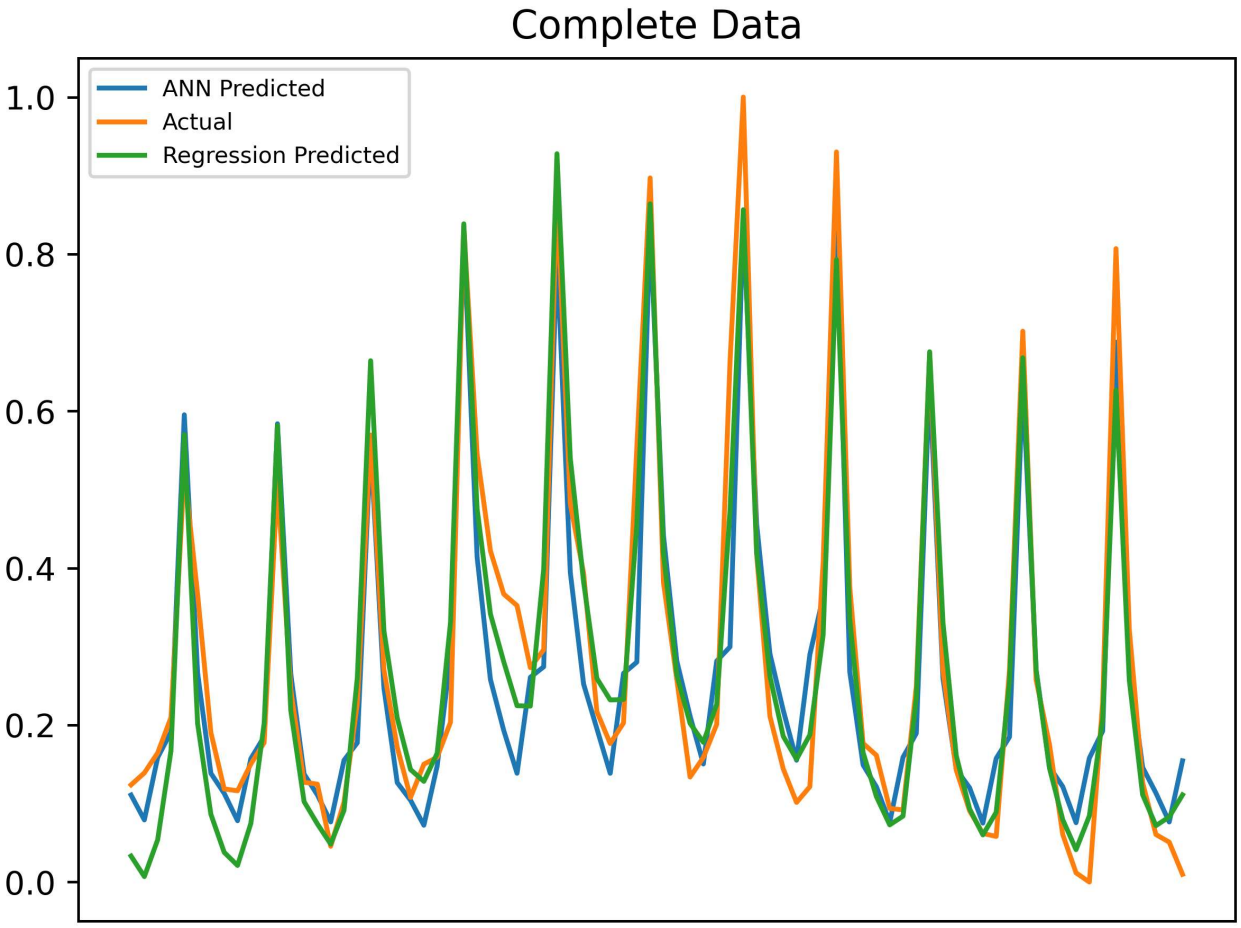


In [469…

```
plt.figure(dpi=441)
plt.plot(np.arange(24), pred_test_ann, label="ANN Predicted")
plt.plot(np.arange(24), y_test, label='Actual')
plt.plot(np.arange(24), pred_test, label = "Regression Predicted")
plt.legend(loc='upper center', fontsize= 7)
plt.title("Test Data")
plt.xticks([])
plt.show()
```

In [466…

```
plt.figure(dpi=441)
plt.plot(np.arange(80), pred_all_ann, label="ANN Predicted")
plt.plot(np.arange(80), y, label='Actual')
plt.plot(np.arange(80), pred_all, label = "Regression Predicted")
plt.legend(loc='upper left', fontsize=7)
plt.title("Complete Data")
plt.xticks([])
plt.show()
```


Complete Data
ANN Predicted
Actual
Regression Predicted
1.0
0.8
0.6
0.4
0.2
0.0

THE END