In [144]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

%matplotlib notebook

import seaborn as sns
```

In [145]:
```
plt.style.use("seaborn-colorblind")
```

In [146]:
```
data = pd.read_csv("C:/Users/VARUN/Desktop/AirPollution/Dataset/cpcb_dly_aq_telangana-2014.csv")
```

In [147]:
```
data.head()
```

Out[147]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	Location of Monitoring Station	Agency	Type of Location	SO2	NO2	RSPM/PM10	PM 2.5
0	95	01-03-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	26.0	217.0	NaN
1	95	01-06-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	27.0	136.0	NaN
2	95	01-09-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	30.0	130.0	NaN
3	95	01-12-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	6.0	24.0	78.0	NaN
4	95	16-01-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	29.0	110.0	NaN

In [148]:
```
dates = ['-'.join(i.split('-')[1:]) for i in data['Sampling Date']]
```

In [150]:
```
data['Sampling Date'] = dates
```

In [151]:
```
for i in range(len(data['City/Town/Village/Area'])):
    if data['City/Town/Village/Area'][i] != 'Hyderabad':
        data.drop(i, inplace = True)
```

In [152]:
```
data.head()
```

Out[152]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	Location of Monitoring Station	Agency	Type of Location	SO2	NO2	RSPM/PM10	PM 2.5
0	95	03-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	26.0	217.0	NaN
1	95	06-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	27.0	136.0	NaN
2	95	09-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	30.0	130.0	NaN
3	95	12-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	6.0	24.0	78.0	NaN
4	95	01-14	Telangana	Hyderabad	C.I.T.D., Balanagar, Plot no. A1 to A8, IDA, H...	Andhra Pradesh State Pollution Control Board	Industrial Area	5.0	29.0	110.0	NaN

In [153]:
```
data = data.groupby("Sampling Date").mean().drop("PM 2.5", axis=1)
data
```

Out[153]:

Sampling Date	Stn Code	SO2	NO2	RSPM/PM10
01-14	302.698113	4.568627	21.823529	95.981132
02-14	289.620690	5.275862	22.724138	90.224138
03-14	349.033898	5.406780	24.177966	105.008475
04-14	278.619048	5.400000	21.650000	106.793651
05-14	289.603448	4.793103	23.396552	112.155172
06-14	348.500000	5.110294	25.205882	97.227941
07-14	289.732143	4.264151	23.471698	72.125000
08-14	289.750000	4.836364	22.563636	74.200000
09-14	361.536585	4.723577	25.585366	96.373984
10-14	319.754098	5.327869	24.344262	88.540984
11-14	343.916667	4.671756	25.610687	101.924242
12-14	292.641791	5.646154	23.169231	115.149254

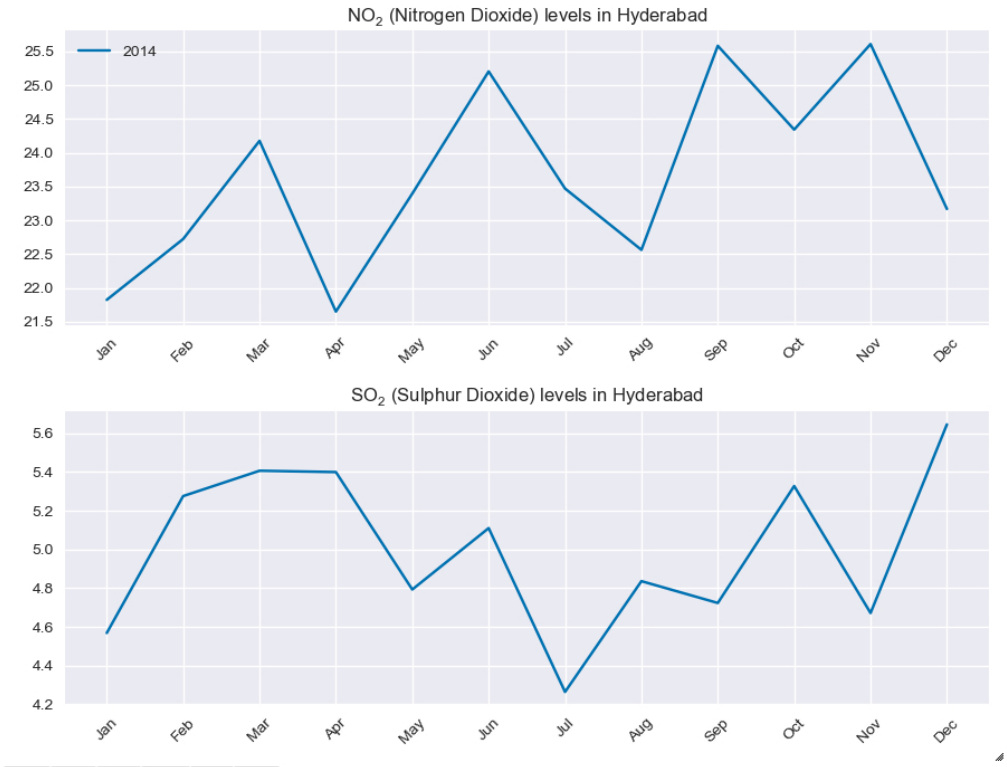
In [154]:
```
plt.figure(figsize=(9, 7))
plt.subplot(211)
plt.cla()
plotter = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun','Jul', 'Aug', 'Sep',
           'Oct', 'Nov', 'Dec']
x = np.arange(0,len(data["NO2"]),1)
ax = plt.gca()
ax.plot(x, data["NO2"])
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.set_xticks(x)
```

```
plt.xticks(rotation=45)
plt.subplots_adjust(bottom=0.2)
plt.legend([2014],loc=2)
plt.title("$\mathregular{NO_2}$ (Nitrogen Dioxide) levels in Hyderabad")
ax.set_xticklabels(plotter)

plt.subplot(212)
plt.cla()
plotter = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun','Jul', 'Aug', 'Sep',
           'Oct', 'Nov', 'Dec']
x = np.arange(0,len(data["SO2"]),1)
ax = plt.gca()
ax.plot(x, data["SO2"])
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.set_xticks(x)
plt.xticks(rotation=45)
plt.title("$\mathregular{SO_2}$ (Sulphur Dioxide) levels in Hyderabad")
ax.set_xticklabels(plotter)

plt.tight_layout()
plt.show()
```

Figure 1



In [131]:
```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split
```

In [169]:
```
forecast_len=3
```

In [170]:
```
x=np.array(data.drop(['NO2'],1))
x=x[:-forecast_len]

y=np.array(data['NO2'])
y=y[:-forecast_len]

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)

print(len(x))

svr_rbf=SVR(kernel='rbf',C=1e3,gamma=0.1)
svr_rbf.fit(x_train,y_train)
```

```
9
```

Out[170]:
```
SVR(C=1000.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma=0.1,
  kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

In [171]:
```
svr_rbf_confidence=svr_rbf.score(x_test,y_test)
print(f"SVR Confidence: {round(svr_rbf_confidence*100,2)}%")
```

```
SVR Confidence: -37.52%
```

In [172]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[172]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [173]:
```
lr_confidence=lr.score(x_test,y_test)
print(f"Linear Regression Confidence: {round(lr_confidence*100,2)}%")
```

```
Linear Regression Confidence: 95.05%
```

In [174]:
```
x_forecast = np.array(data.drop(['NO2'],1))[-forecast_len:]
print(x_forecast)
```

```
[[319.75409836   5.32786885  88.54098361]
 [343.91666667   4.67175573 101.92424242]
 [292.64179104   5.64615385 115.14925373]]
```

In [175]:
```
# Print linear regression model predictions for the next '30' days
lr_prediction = lr.predict(x_forecast)
print(lr_prediction)
# Print support vector regressor model predictions for the next '30' days
svm_prediction = svr_rbf.predict(x_forecast)
print(svm_prediction)
```

```
[23.64899562 24.57528107 22.27675786]
[23.1919205  23.24072659 23.20775192]
```

In [176]:
```
lr=LinearRegression(normalize=True)
lr.fit(x_train,y_train)
lr_confidence=lr.score(x_test,y_test)
print(f"Linear Regression Confidence: {round(lr_confidence*100,2)}%")

from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()

from sklearn import neighbors
knn = neighbors.KNeighborsClassifier(n_neighbors=5)

from sklearn.decomposition import PCA
pca = PCA(n_components=0.95)

from sklearn.cluster import KMeans
k_means = KMeans(n_clusters=3, random_state=0)
```

```
Linear Regression Confidence: 95.05%
```

In [ ]: