ANALYSIS FLOW PROCEDURE

In [1]:

```
import pandas as pd
import numpy as np
import sklearn
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.kernel_ridge import KernelRidge
import matplotlib
from matplotlib import pyplot as pl
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score,
mean_squared_error,mean_absolute_error,median_absolute_error
```

DESC: Importing csv file containing datasets for the project

In [2]:

```
d=pd.read_csv('weather-prediction-dataset.csv')
```

DESC: Finding the correlation of different features and sort them in descending order

In [3]:

```
cor=d.corr()[['meantempm']].sort_values(by='meantempm',ascending=False)
```

DESC: Creating a new dataframe and inserting the selected features into it

In [4]:

```
List=['maxtempm','mintempm','meantempm_1','maxtempm_1','mintempm_1',
      'mindewptm_1','meandewptm_1','maxdewptm_1','meantempm_2'
      ,'maxtempm_2','meantempm_3','mintempm_2','mindewptm_2','meandewptm_2',
      'maxdewptm_2','mintempm_3','meandewptm_3',
      'mindewptm_3','maxtempm_3','maxdewptm_3']
DFnew=pd.DataFrame()
for n in List:
    DFnew[n]=d[n]
DFnew['LABEL_MEAN_TEMP_M']=d['meantempm']
```

DESC: Converting DFnew into array and storing it in 'weather' slicing the features and labels

In [5]:

```
weather=np.array(DFnew)
features=weather[:,:20]
label=weather[:,20]
```

DESC: Shuffling the features and label

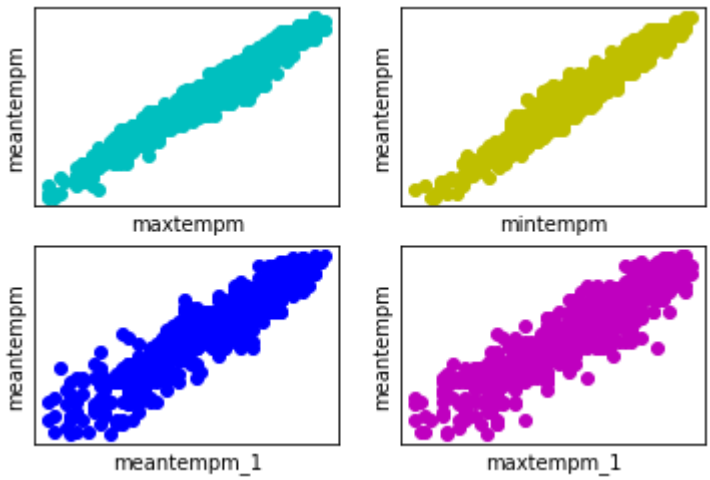
In [6]:

```
m=int(3/4*len(features))
c=len(features)
shuf=np.random.permutation(c)
features=features[shuf]
label=label[shuf]
```

DESC: Plotting graphs of different features

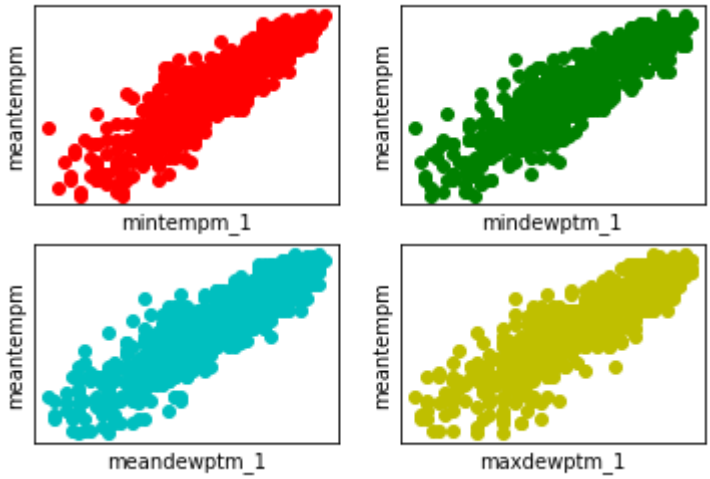
In [7]:

```
j=['og','oc','oy','ob','om','or']
for i in range(1,5):
    x=features[:,i-1]
    x_train=x[:m]
    y_train=label[:m]
    pl.subplot(2,2,i)
    pl.plot(x_train,y_train,j[i%6])
    pl.xticks(())
    pl.yticks(())
    pl.xlabel(List[i-1])
    pl.ylabel('meantempm')
pl.show()
```



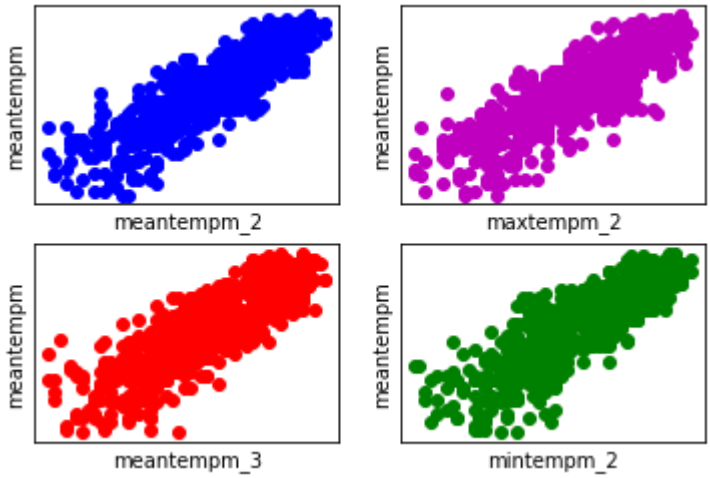
In [8]:

```
for i in range(5,9):
    x=features[:,i-1]
    x_train=x[:m]
    y_train=label[:m]
    pl.subplot(2,2,i%5+1)
    pl.plot(x_train,y_train,j[i%6])
    pl.xticks(())
    pl.yticks(())
    pl.xlabel(List[i-1])
    pl.ylabel('meantempm')
pl.show()
```



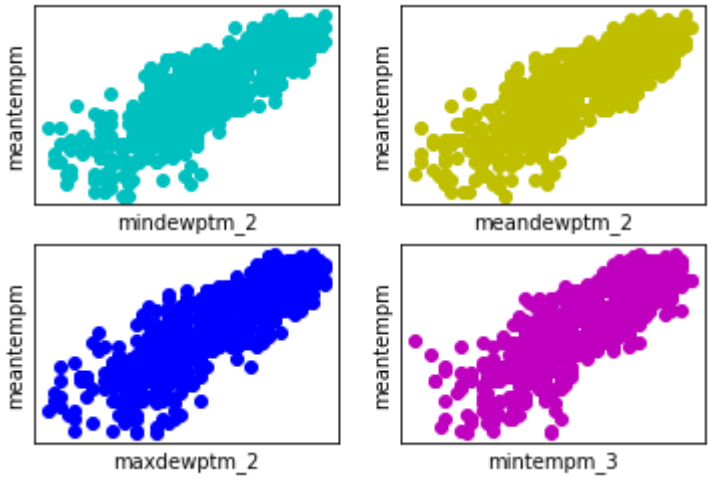
In [9]:

```
for i in range(9,13):
    x=features[:,i-1]
    x_train=x[:m]
    y_train=label[:m]
    pl.subplot(2,2,i%9+1)
    pl.plot(x_train,y_train,j[i%6])
    pl.xticks(())
    pl.yticks(())
    pl.xlabel(List[i-1])
    pl.ylabel('meantempm')
pl.show()
```



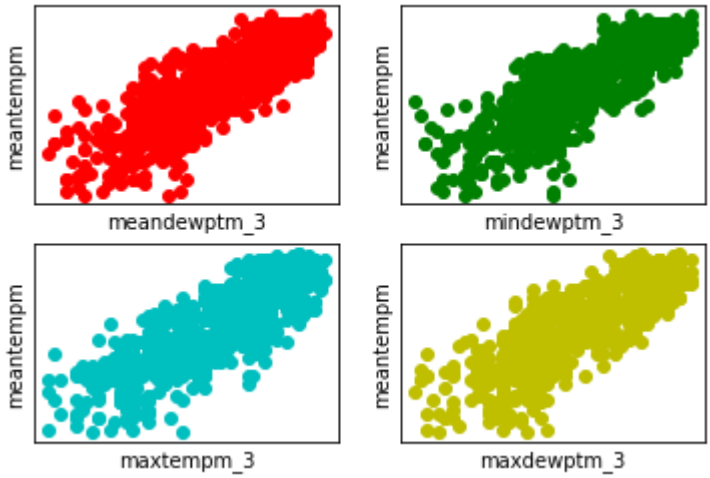
In [10]:

```
for i in range(13,17):
    x=features[:,i-1]
    x_train=x[:m]
    y_train=label[:m]
    pl.subplot(2,2,i%13+1)
    pl.plot(x_train,y_train,j[i%6])
    pl.xticks(())
    pl.yticks(())
    pl.xlabel(List[i-1])
    pl.ylabel('meantempm')
pl.show()
```



In [11]:

```
for i in range(17,21):
    x=features[:,i-1]
    x_train=x[:m]
    y_train=label[:m]
    pl.subplot(2,2,i%17+1)
    pl.plot(x_train,y_train,j[i%6])
    pl.xticks(())
    pl.yticks(())
    pl.xlabel(List[i-1])
    pl.ylabel('meantempm')
pl.show()
```



DESC: Spliting features and label using functions

In [13]:

```
train_features,test_features,train_label,test_label
=train_test_split(features,label,test_size=0.25)
```

DESC: Training our model and evaluating errors and accuracy using linear regression

In [15]:

```
model=LinearRegression()
model.fit(train_features,train_label)
pred_y=model.predict(test_features)
accuracy=model.score(test_features,test_label)
ms=mean_squared_error(test_label,pred_y)
l1=[]
print('linear_mean_square',ms)
fs=r2_score(test_label,pred_y)
print('linear_r2_score',fs)
mean=sklearn.metrics.mean_absolute_error(test_label,pred_y)
median=sklearn.metrics.median_absolute_error(test_label,pred_y)
print('linear_mean',mean)
print('linear_median',median)
l1=[ms,fs,mean,median]
```

```
linear_mean_square 0.1486190521921545
linear_r2_score 0.9987931795338042
linear_mean 0.3248428066682208
linear_median 0.2837403954291027
```

DESC: Training our model and evaluating errors and accuracy using Support Vector Regression(SVR) where kernel='linear'

In [16]:

```
clf=SVR(kernel="linear")
clf.fit(train_features,train_label)
l2=[]
pred_y=clf.predict(test_features)
accuracy1=clf.score(test_features,test_label)
ms=mean_squared_error(test_label,pred_y)
print('svr_mean_square',ms)
fs=r2_score(test_label,pred_y)
print('svr_r2_score',fs)
mean1=sklearn.metrics.mean_absolute_error(test_label,pred_y)
print('svr_mean',mean1)
median1=sklearn.metrics.median_absolute_error(test_label,pred_y)
print('svr_median',median1)
l2=[ms,fs,mean1,median1]
```

```
svr_mean_square 0.15336856893041967
svr_r2_score 0.9987546123789224
svr_mean 0.3268485549213921
svr_median 0.29565166033502166
```

DESC: Training our model and evaluating errors and accuracy using KernelRidge

In [17]:

```
clf1=KernelRidge()
clf1.fit(train_features,train_label)
l3=[]
pred_y=clf1.predict(test_features)
ms=mean_squared_error(test_label,pred_y)
print('kernelRidge_mean_square',ms)
fs=r2_score(test_label,pred_y)
print('kernelRidge_r2_score',fs)
mean2=sklearn.metrics.mean_absolute_error(test_label,pred_y)
print('kernelRidge_mean',mean2)
median2=sklearn.metrics.median_absolute_error(test_label,pred_y)
print('kernelRidge_median',median2)
l3=[ms,fs,mean2,median2]
```

```
kernelRidge_mean_square 0.14893615999691326
kernelRidge_r2_score 0.9987906045463908
kernelRidge_mean 0.3248585718188533
kernelRidge_median 0.2860633811196749
```

DESC: Training our model and evaluating errors and accuracy using SVR where Kernel='rbf'

In [19]:

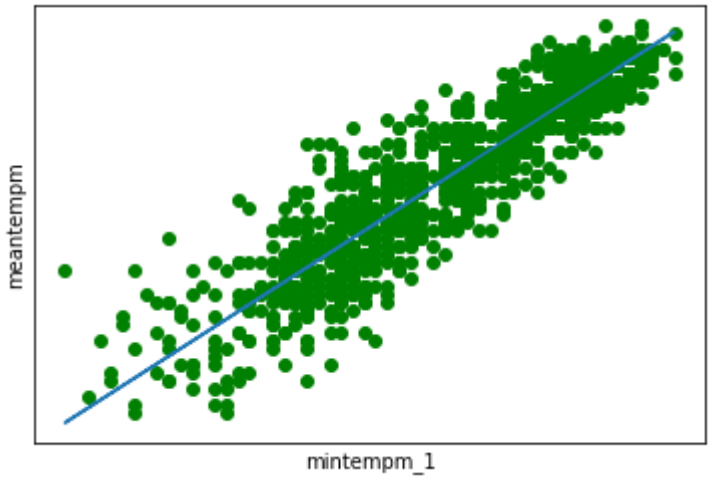
```
clf2=SVR()
clf2.fit(train_features,train_label)
l4=[]
pred_y=clf2.predict(test_features)
ms=mean_squared_error(test_label,pred_y)
print('svr_rbf_mean_square',ms)
fs=r2_score(test_label,pred_y)
print('svr_rbf_r2_score',fs)
mean3=sklearn.metrics.mean_absolute_error(test_label,pred_y)
print('svr_rbf_mean',mean3)
median3=sklearn.metrics.median_absolute_error(test_label,pred_y)
print('svr_rbf_median',median3)
l4=[ms,fs,mean3,median3]
```

```
svr_rbf_mean_square 122.21946227148158
svr_rbf_r2_score 0.007550201262416234
svr_rbf_mean 9.079954618817908
svr_rbf_median 8.26104428817596
```

DESC: Classifying the plotted data of a particular feature (here 4th column) using SVR(kernel='linear')

In [20]:

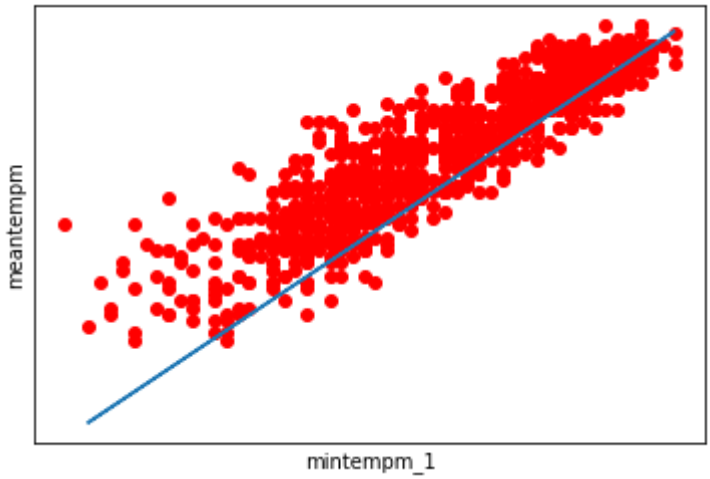
```
x2=features[:,np.newaxis,4]
y2=label
pl.plot(x2,y2,'og')
pl.xticks(())
pl.yticks(())
pl.xlabel('mintempm_1')
pl.ylabel('meantempm')
train_x2,test_x2,train_y2,test_y2=train_test_split(x2,y2,test_size=0.25)
svr1=SVR(kernel="linear")
svr1.fit(train_x2,train_y2)
pred_y2=svr1.predict(test_x2)
pl.plot(test_x2,pred_y2)
pl.show()
```



DESC: Classifying the plotted data of a particular feature (here 4th coumn) using KernelRidge

In [21]:

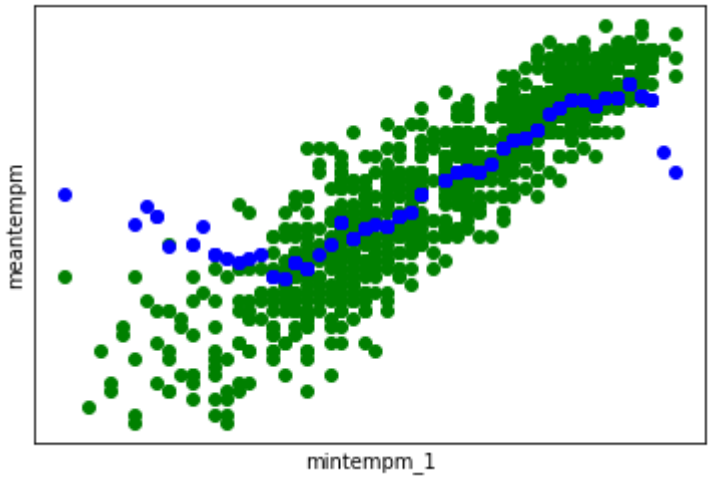
```
x3=features[:,np.newaxis,4]
y3=label
pl.plot(x3,y3,'or')
pl.xticks(())
pl.yticks(())
pl.xlabel('mintempm_1')
pl.ylabel('meantempm')
train_x3,test_x3,train_y3,test_y3=train_test_split(x3,y3,test_size=0.25)
svr1=KernelRidge()
svr1.fit(train_x3,train_y3)
pred_y3=svr1.predict(test_x3)
pl.plot(test_x3,pred_y3)
pl.show()
```



DESC: Classifying the plotted data of a particular feature (here 4th coumn) using SVR(kernel='rbf)

In [22]:

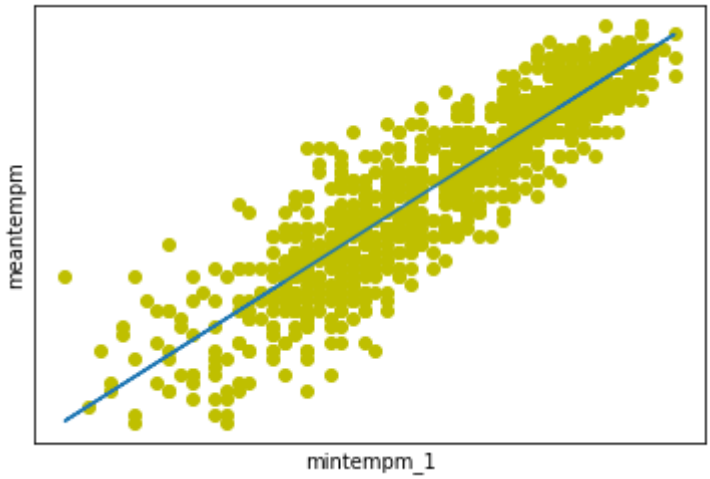
```
x4=features[:,np.newaxis,4]
y4=label
pl.plot(x4,y4,'og')
pl.xticks(())
pl.yticks(())
pl.xlabel('mintempm_1')
pl.ylabel('meantempm')
train_x4,test_x4,train_y4,test_y4=train_test_split(x4,y4,test_size=0.25)
svr4=SVR()
svr4.fit(train_x4,train_y4)
pred_y4=svr4.predict(test_x4)
pl.plot(test_x4,pred_y4,'ob')
pl.show()
```



DESC: Classifying the plotted data of a particular feature (here 4th coumn) using LinearRegression

In [23]:

```
x5=features[:,np.newaxis,4]
y5=label
pl.plot(x5,y5,'oy')
pl.xticks(())
pl.yticks(())
pl.xlabel('mintempm_1')
pl.ylabel('meantempm')
train_x5,test_x5,train_y5,test_y5=train_test_split(x5,y5,test_size=0.25)
lr=LinearRegression()
lr.fit(train_x5,train_y5)
pred_y5=lr.predict(test_x5)
pl.plot(test_x5,pred_y5)
pl.show()
```



DESC: Creating DataFrame on the above data to compare each algorithm

In [24]:

```
INDEX=['mean square error','r2 score','mean absolute error','median absolute error']
DFtab=pd.DataFrame(index=INDEX)
DFtab['Linear Regression']=l1
DFtab['SVR Linear']=l2
DFtab['Kernel']=l3
DFtab['SVR Rbf']=l4
print(DFtab)
```

```
                      Linear Regression  SVR Linear    Kernel     SVR Rbf
mean square error                0.148619    0.153369  0.148936  122.219462
r2 score                         0.998793    0.998755  0.998791    0.007550
mean absolute error              0.324843    0.326849  0.324859    9.079955
median absolute error            0.283740    0.295652  0.286063    8.261044
```

In [ ]: