```
# importing the pandas module for
# data frame
import pandas as pd


# load the data set into train variable.
train = pd.read_csv('/content/cars.csv')

# display top 5 values of data set
train.head()
```

	DateTime	Vehicles
0	1/11/2015 0:00	2
1	1/12/2015 0:00	10
2	1/13/2015 0:00	4
3	1/14/2015 0:00	22
4	1/15/2015 0:00	16

```
# function to get all data from time stamp

# get date
def get_dom(dt):
    return dt.day

# get week day
def get_weekday(dt):
    return dt.weekday()

# get hour
def get_hour(dt):
    return dt.hour

# get year
def get_year(dt):
    return dt.year

# get month
def get_month(dt):
    return dt.month

# get year day
def get_dayofyear(dt):
    return dt.dayofyear

# get year week
def get_weekofyear(dt):
    return dt.weekofyear


train['DateTime'] = train['DateTime'].map(pd.to_datetime)
train['date'] = train['DateTime'].map(get_dom)
train['weekday'] = train['DateTime'].map(get_weekday)
train['hour'] = train['DateTime'].map(get_hour)
train['month'] = train['DateTime'].map(get_month)
train['year'] = train['DateTime'].map(get_year)
train['dayofyear'] = train['DateTime'].map(get_dayofyear)
train['weekofyear'] = train['DateTime'].map(get_weekofyear)

# display
train.head()
```

	DateTime	Vehicles	date	weekday	hour	month	year	dayofyear	weekofyear
0	2015-01-11	2	11.0	6.0	0.0	1.0	2015.0	11.0	2.0

```
train.to_csv('file1.csv')
```

2	2015-01-13	4	13.0	1.0	0.0	1.0	2015.0	13.0	3.0

```
# display top 5 values of data set
train.head()


# there is no use of DateTime module
# so remove it
train = train.drop("DateTime", axis=0)

# separating class label for training the data
train1 = train.drop(['Vehicles'], axis=1)

# class label is stored in target
target = train['Vehicles']

print(train1.head())
target.head()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-27-3e225142577f> in <cell line: 3>()
      1 # there is no use of DateTime module
      2 # so remove it
----> 3 train = train.drop("DateTime", axis=0)
      4
      5 # separating class label for training the data

                                5 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/indexes/base.py in drop(self, labels, errors)
   6932         if mask.any():
   6933             if errors != "ignore":
-> 6934                 raise KeyError(f"{list(labels[mask])} not found in axis")
   6935             indexer = indexer[~mask]
   6936         return self.delete(indexer)

KeyError: "['DateTime'] not found in axis"
```

SEARCH STACK OVERFLOW

```
target = pd.read_csv('/content/file1.csv')

# separating class label for training the data
train1 = train.drop('Vehicles', axis=1)

# class label is stored in target
target = train['Vehicles']

print(train1.head())
target.head()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
```

```
train1 = pd.read_csv('/content/train1.csv')
train1 = train1.fillna(train1.mean())
target = pd.read_csv('/content/target.csv')
#importing Random forest
from sklearn.ensemble import RandomForestRegressor

#defining the RandomForestRegressor
m1=RandomForestRegressor()

m1.fit(train1,target)
#testing
m1.predict([[11,6,0,1,2015,11,2]])
```

```
<ipython-input-36-1cd535047729>:10: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the
  m1.fit(train1,target)
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but RandomForestRegressor
  warnings.warn(
array([5.73])
```

```
print(m1.predict([[11,6,0,1,2015,11,2]]))
```

```
[5.73]
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but RandomForestRegressor
  warnings.warn(
```