In [1]:
```
import pandas as pd
df = pd.read_excel("C:/Users/hp/OneDrive/Desktop/Florien Amondi_FA/Absenteeism_at_work.xls")
```

In [2]:
```
df.head()
```

Out[2]:

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	...	Disciplinary failure	Education	Son	Social drinker
0	11	26	7	3	1	289	36	13	33	239554	...	0	1	2	1
1	36	0	7	3	1	118	13	18	50	239554	...	1	1	1	1
2	3	23	7	4	1	179	51	18	38	239554	...	0	1	0	1
3	7	7	7	5	1	279	5	14	39	239554	...	0	1	2	1
4	11	23	7	5	1	289	36	13	33	239554	...	0	1	2	1

5 rows × 21 columns

In [3]:
```
df.columns
```

Out[3]:
```
Index(['ID', 'Reason for absence', 'Month of absence', 'Day of the week',
       'Seasons', 'Transportation expense', 'Distance from Residence to Work',
       'Service time', 'Age', 'Work load Average/day ', 'Hit target',
       'Disciplinary failure', 'Education', 'Son', 'Social drinker',
       'Social smoker', 'Pet', 'Weight', 'Height', 'Body mass index',
       'Absenteeism time in hours'],
      dtype='object')
```

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 740 entries, 0 to 739
Data columns (total 21 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   ID                               740 non-null    int64
 1   Reason for absence               740 non-null    int64
 2   Month of absence                 740 non-null    int64
 3   Day of the week                  740 non-null    int64
 4   Seasons                          740 non-null    int64
 5   Transportation expense           740 non-null    int64
 6   Distance from Residence to Work  740 non-null    int64
 7   Service time                     740 non-null    int64
 8   Age                              740 non-null    int64
 9   Work load Average/day            740 non-null    int64
 10  Hit target                       740 non-null    int64
 11  Disciplinary failure             740 non-null    int64
 12  Education                        740 non-null    int64
 13  Son                              740 non-null    int64
 14  Social drinker                   740 non-null    int64
 15  Social smoker                    740 non-null    int64
 16  Pet                              740 non-null    int64
 17  Weight                           740 non-null    int64
 18  Height                           740 non-null    int64
 19  Body mass index                  740 non-null    int64
 20  Absenteeism time in hours        740 non-null    int64
dtypes: int64(21)
memory usage: 121.5 KB
```

In [8]:
```
df.dtypes
```

Out[8]:
```
ID                               int64
Reason for absence               int64
Month of absence                 int64
Day of the week                  int64
Seasons                          int64
Transportation expense           int64
Distance from Residence to Work  int64
Service time                     int64
Age                              int64
Work load Average/day            int64
Hit target                       int64
Disciplinary failure             int64
Education                        int64
Son                              int64
Social drinker                   int64
Social smoker                    int64
Pet                              int64
Weight                           int64
Height                           int64
Body mass index                  int64
Absenteeism time in hours        int64
dtype: object
```

In [6]:
```
df.describe()
```

Out[6]:

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	...	D
count	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	...	7
mean	18.017568	19.216216	6.324324	3.914865	2.544595	221.329730	29.631081	12.554054	36.450000	271490.235135	...	
std	11.021247	8.433406	3.436287	1.421675	1.111831	66.952223	14.836788	4.384873	6.478772	39058.116188	...	
min	1.000000	0.000000	0.000000	2.000000	1.000000	118.000000	5.000000	1.000000	27.000000	205917.000000	...	
25%	9.000000	13.000000	3.000000	3.000000	2.000000	179.000000	16.000000	9.000000	31.000000	244387.000000	...	
50%	18.000000	23.000000	6.000000	4.000000	3.000000	225.000000	26.000000	13.000000	37.000000	264249.000000	...	
75%	28.000000	26.000000	9.000000	5.000000	4.000000	260.000000	50.000000	16.000000	40.000000	294217.000000	...	
max	36.000000	28.000000	12.000000	6.000000	4.000000	388.000000	52.000000	29.000000	58.000000	378884.000000	...	

8 rows × 21 columns

In [7]:
```
# Rescale data (between 0 and 1)
from numpy import set_printoptions
from sklearn.preprocessing import MinMaxScaler
array = df.values
# separate array into input and output components
X = array[:,0:20]
Y = array[:,20]
scaler = MinMaxScaler(feature_range=(0, 1))
rescaledX = scaler.fit_transform(X)
# summarize transformed data
set_printoptions(precision=3)
print(rescaledX[0:5,:])
```

```
[[0.286 0.929 0.583 0.25  0.    0.633 0.66  0.429 0.194 0.194 0.842 0.
  0.    0.5   1.    0.    0.125 0.654 0.273 0.579]
 [1.    0.    0.583 0.25  0.    0.    0.17  0.607 0.742 0.194 0.842 1.
  0.    0.25  1.    0.    0.    0.808 0.455 0.632]
 [0.057 0.821 0.583 0.5   0.    0.226 0.979 0.607 0.355 0.194 0.842 0.
  0.    0.    1.    0.    0.    0.635 0.212 0.632]
 [0.171 0.25  0.583 0.75  0.    0.596 0.    0.464 0.387 0.194 0.842 0.
  0.    0.5   1.    1.    0.    0.231 0.152 0.263]
 [0.286 0.821 0.583 0.75  0.    0.633 0.66  0.429 0.194 0.194 0.842 0.
  0.    0.5   1.    0.    0.125 0.654 0.273 0.579]]
```

In [8]:
```
rescaledXDF = pd.DataFrame(rescaledX,columns=['ID', 'Reason for absence', 'Month of absence', 'Day of the week'
```

In [9]:
```
rescaledXDF.head()
```

Out[9]:

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target	Disciplinary failure	Educati
0	0.285714	0.928571	0.583333	0.25	0.0	0.633333	0.659574	0.428571	0.193548	0.194471	0.842105	0.0	
1	1.000000	0.000000	0.583333	0.25	0.0	0.000000	0.170213	0.607143	0.741935	0.194471	0.842105	1.0	
2	0.057143	0.821429	0.583333	0.50	0.0	0.225926	0.978723	0.607143	0.354839	0.194471	0.842105	0.0	
3	0.171429	0.250000	0.583333	0.75	0.0	0.596296	0.000000	0.464286	0.387097	0.194471	0.842105	0.0	
4	0.285714	0.821429	0.583333	0.75	0.0	0.633333	0.659574	0.428571	0.193548	0.194471	0.842105	0.0	

In [10]:
```
rescaledXDF.describe()
```

Out[10]:

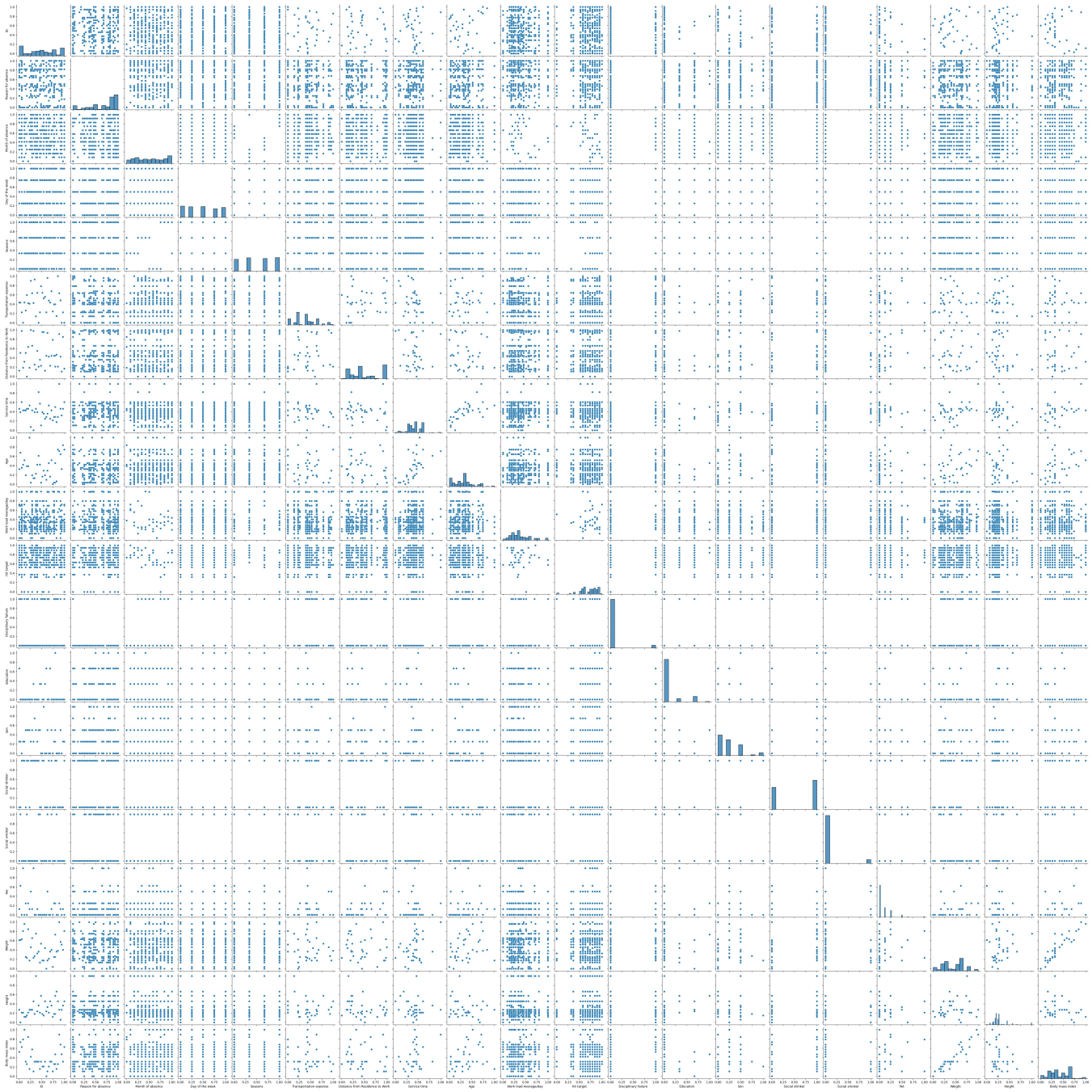
	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit ta
count	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000
mean	0.486216	0.686293	0.527027	0.478716	0.514865	0.382703	0.524066	0.412645	0.304839	0.379108	0.715
std	0.314893	0.301193	0.286357	0.355419	0.370610	0.247971	0.315676	0.156603	0.208993	0.225813	0.198
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	0.228571	0.464286	0.250000	0.250000	0.333333	0.225926	0.234043	0.285714	0.129032	0.222412	0.631
50%	0.485714	0.821429	0.500000	0.500000	0.666667	0.396296	0.446809	0.428571	0.322581	0.337244	0.736
75%	0.771429	0.928571	0.750000	0.750000	1.000000	0.525926	0.957447	0.535714	0.419355	0.510502	0.842
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000

In [11]:
```
import seaborn as sns
df.columns
```

Out[11]:
```
Index(['ID', 'Reason for absence', 'Month of absence', 'Day of the week',
       'Seasons', 'Transportation expense', 'Distance from Residence to Work',
       'Service time', 'Age', 'Work load Average/day ', 'Hit target',
       'Disciplinary failure', 'Education', 'Son', 'Social drinker',
       'Social smoker', 'Pet', 'Weight', 'Height', 'Body mass index',
       'Absenteeism time in hours'],
      dtype='object')
```

In [12]:
```
sns.pairplot(rescaledXDF)
```

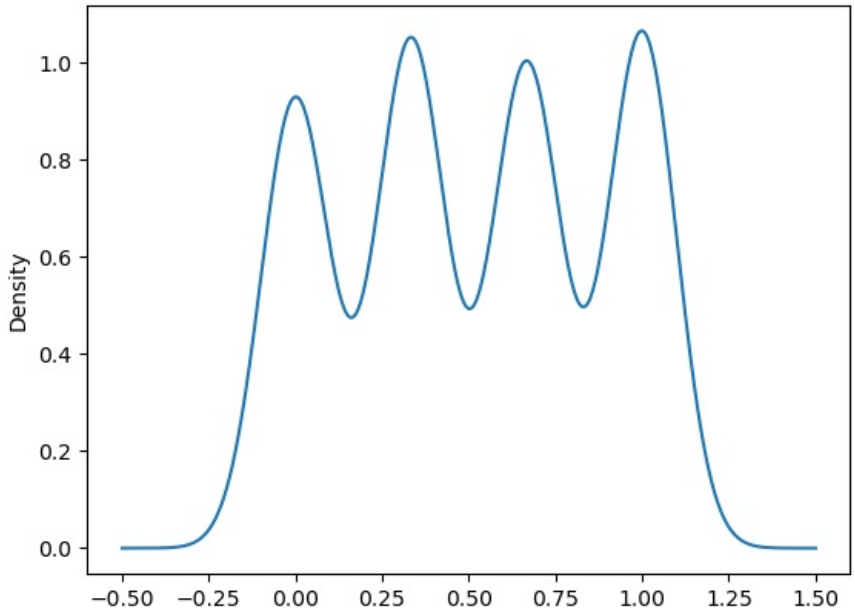
Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x1aaf37ceee0>
```



In [ ]:
```
```

In [13]:
```
rescaledXDF.Seasons.plot(kind = "density")
```

Out[13]:
```
<AxesSubplot:ylabel='Density'>
```



In [ ]:

In [14]:
```
from sklearn.preprocessing import Normalizer
from pandas import read_csv
from numpy import set_printoptions
array = rescaledXDF.values
# separate array into input and output components
X = array[:,0:19]
Y = array[:,19]
scaler = Normalizer().fit(X)
normalizedX = scaler.transform(X)
# summarize transformed data
set_printoptions(precision=3)
print(normalizedX[0:5,:])
```

```
[[0.129 0.419 0.263 0.113 0.    0.286 0.297 0.193 0.087 0.088 0.38  0.
  0.    0.225 0.451 0.    0.056 0.295 0.123]
 [0.408 0.    0.238 0.102 0.    0.    0.069 0.247 0.302 0.079 0.343 0.408
  0.    0.102 0.408 0.    0.    0.329 0.185]
 [0.026 0.369 0.262 0.224 0.    0.101 0.439 0.272 0.159 0.087 0.378 0.
  0.    0.    0.449 0.    0.    0.285 0.095]
 [0.078 0.114 0.267 0.343 0.    0.272 0.    0.212 0.177 0.089 0.385 0.
  0.    0.228 0.457 0.457 0.    0.105 0.069]
 [0.125 0.359 0.255 0.328 0.    0.277 0.288 0.187 0.085 0.085 0.368 0.
  0.    0.219 0.437 0.    0.055 0.286 0.119]]
```

In [15]:
```
normDF = pd.DataFrame(rescaledXDF,columns=['ID', 'Reason for absence', 'Month of absence', 'Day of the week','S
```

In [16]:
```
normDF.head()
```

Out[16]:

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target	Disciplinary failure	Educati
0	0.285714	0.928571	0.583333	0.25	0.0	0.633333	0.659574	0.428571	0.193548	0.194471	0.842105	0.0	
1	1.000000	0.000000	0.583333	0.25	0.0	0.000000	0.170213	0.607143	0.741935	0.194471	0.842105	1.0	
2	0.057143	0.821429	0.583333	0.50	0.0	0.225926	0.978723	0.607143	0.354839	0.194471	0.842105	0.0	
3	0.171429	0.250000	0.583333	0.75	0.0	0.596296	0.000000	0.464286	0.387097	0.194471	0.842105	0.0	
4	0.285714	0.821429	0.583333	0.75	0.0	0.633333	0.659574	0.428571	0.193548	0.194471	0.842105	0.0	

In [17]:
```
normDF.describe()
```

Out[17]:

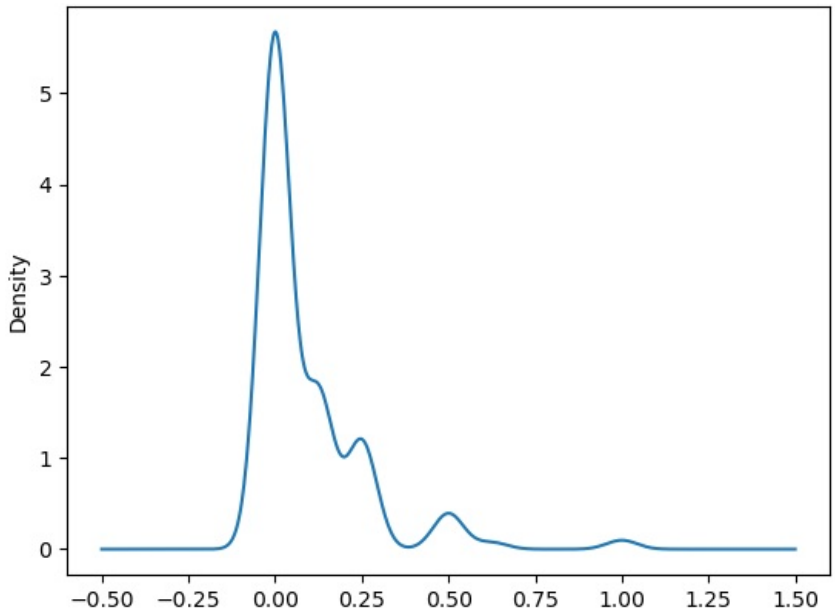
	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit ta
count	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000000	740.000
mean	0.486216	0.686293	0.527027	0.478716	0.514865	0.382703	0.524066	0.412645	0.304839	0.379108	0.715
std	0.314893	0.301193	0.286357	0.355419	0.370610	0.247971	0.315676	0.156603	0.208993	0.225813	0.198
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	0.228571	0.464286	0.250000	0.250000	0.333333	0.225926	0.234043	0.285714	0.129032	0.222412	0.631
50%	0.485714	0.821429	0.500000	0.500000	0.666667	0.396296	0.446809	0.428571	0.322581	0.337244	0.736
75%	0.771429	0.928571	0.750000	0.750000	1.000000	0.525926	0.957447	0.535714	0.419355	0.510502	0.842
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000

In [18]:
```
normDF.columns
```

Out[18]:
```
Index(['ID', 'Reason for absence', 'Month of absence', 'Day of the week',
       'Seasons', 'Transportation expense', 'Distance from Residence to Work',
       'Service time', 'Age', 'Work load Average/day ', 'Hit target',
       'Disciplinary failure', 'Education', 'Son', 'Social drinker',
       'Social smoker', 'Pet', 'Weight', 'Height', 'Body mass index'],
      dtype='object')
```

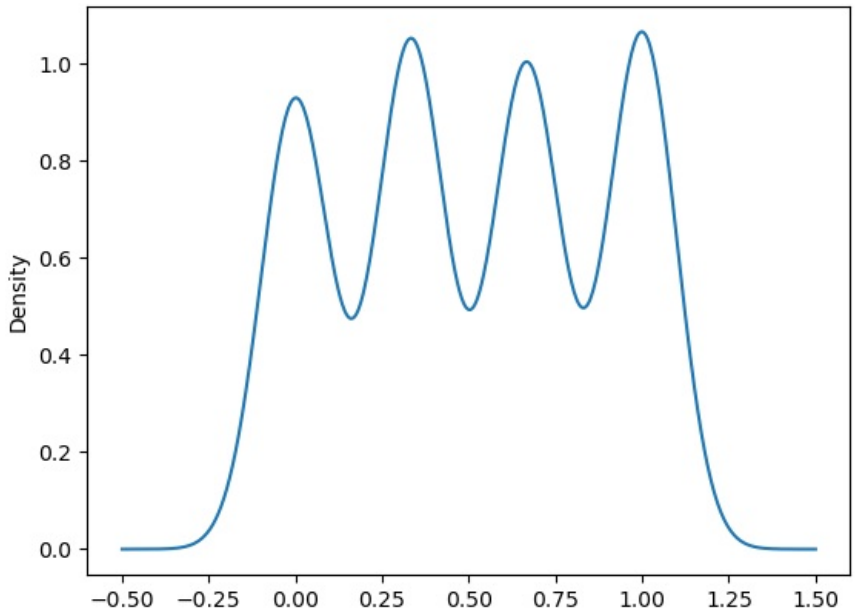
In [19]:
```
import pandas as pd
import matplotlib.pyplot as plt

normDF.Pet.plot(kind = "density")
```

Out[19]:
```
<AxesSubplot:ylabel='Density'>
```



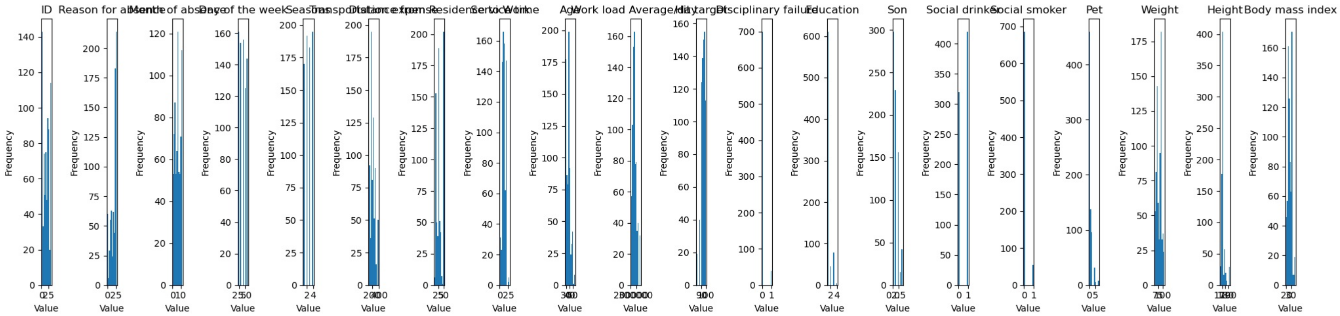
In [20]:
```
normDF.Seasons.plot(kind = "density")
```

Out[20]:
```
<AxesSubplot:ylabel='Density'>
```



In [21]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Load CSV data


# Select the features to create histograms for
hist_features = ['ID', 'Reason for absence', 'Month of absence', 'Day of the week',
       'Seasons', 'Transportation expense', 'Distance from Residence to Work',
       'Service time', 'Age', 'Work load Average/day ', 'Hit target',
       'Disciplinary failure', 'Education', 'Son', 'Social drinker',
       'Social smoker', 'Pet', 'Weight', 'Height', 'Body mass index']

# Create a figure with subplots for each feature
fig, axs = plt.subplots(nrows=1, ncols=len(hist_features), figsize=(20,5))

# Create a histogram for each feature
for i, feature in enumerate(hist_features):
    axs[i].hist(df[feature], bins=10)
    axs[i].set_title(feature)
    axs[i].set_xlabel('Value')
    axs[i].set_ylabel('Frequency')

# Adjust the layout of the subplots
plt.tight_layout()

# Show the plot
plt.show()
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: