Hands-on Project

Question 1: List the dataset(s) you chose for this project from the UCI Machine Learning respository (https://archive.ics.uci.edu/ml/datasets.php).

1. Absenteeism at work Data Set (https://archive.ics.uci.edu/ml/datasets/Absenteeism+at+work (https://archive.ics.uci.edu/ml/datasets/Absenteeism+at+work))

Question 2: Describe the dataset in your own words. How many data points, how many attributes, how many types of attributes, how many classes (if any)? Who collected it? How was it collected?

The dataset contains records of absenteeism at work from July 2007 to July 2010 at a courier company in Brazil.
No. of Observations: 740,
No. of Attributes: 21
Types of attibutes: All numerical, but 10 of the attributes in reality are categorical variables already converted to numerics
Attributes Description:
ID - Unique Id for each employee, a total of 36 unique employees.
Reason for absence - 0 to 28, with each number assuming a value for each reason.
1 - Certain infectious and parasitic diseases
2 - Neoplasms
3 - Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism
4 - Endocrine, nutritional and metabolic diseases
5 - Mental and behavioural disorders
6 - Diseases of the nervous system
7 - Diseases of the eye and adnexa
8 - Diseases of the ear and mastoid process
9 - Diseases of the circulatory system
10 - Diseases of the respiratory system
11 - Diseases of the digestive system
12 - Diseases of the skin and subcutaneous tissue
13 - Diseases of the musculoskeletal system and connective tissue
14 - Diseases of the genitourinary system
15 - Pregnancy, childbirth and the puerperium
16 - Certain conditions originating in the perinatal period
17 - Congenital malformations, deformations and chromosomal abnormalities
18 - Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified
19 - Injury, poisoning and certain other consequences of external causes
20 - External causes of morbidity and mortality
21 - Factors influencing health status and contact with health services.
22 - patient follow-up
23 - medical consultation
24 - blood donation
25 - laboratory examination
26 - unjustified absence
27 - physiotherapy
28 - dental consultation
Month of absence - Month of absence from 0 to 12
Day of the week - Monday through Friday represented as 2 to 6
Seasons - 4 seasons 1 to 4
Transportation expense - Expenses when travellinng to and from work
Distance from Residence to Work - Distance from Residence to Work
Service time - Years of experience in that company
Age - Age
Work load Average/day - Average workload per day
Hit target - Hit target
Disciplinary failure - Binary variable suggesting any disciplinary failures reported previously for the employee
Education - high school (1), graduate (2), postgraduate (3), master and doctor (4)

Son - # of childeren
Social drinker - Binary Variable 0 if a non-drinker 1 otherwise
Social smoker - Binary Variable 0 if a non-snoker 1 otherwise
Pet - # of pets
Weight - Weight of the employee
Height - Height of the employee
Body mass index - BMI of the employee
Absenteeism time in hours - # of hours absent

Classes: None

Question 3: What is your goal? Specifically, what insights do you want to learn from this data. Please be aware that clustering, classification, or itemset mining are not 'insights'. These are data mining tasks. Insights are relevant to the domain from which the data is generated.

Goal: The primary goal is to come up with attributes which differentiate high absentees from low absentees, we then based on this analysis would like to suggest some measures to the management of the company that'd help them reduce absenteeism at work. We first transform the dataset into a new summary dataset with one row for one unique employee which would contain summary of that employee for all atrributes. After transforming the dataset now we divide the dataset into two class, marking people with low absenteeism (i.e total absenteeism for 3 years is less than 200) as '0' and people with high absenteeism (i.e total absenteeism for 3 years is greater than or equal to 200) as '1'. we also additionally propose a machine learning model which would predict the likelihood of a given person based on the attribute, since we're unlikely to have company specific data for any new employee, our model (prone to all kinds of prejudice because of the data itself) will most likely predict based on the demographics. If time permits we'd also like to analyze the causes behind such results.

Question 4: List the data mining task(s) and the specific algorithms you want to perform on this data. Do not pick the tasks listed in the 'Default Task' column on the UCI page.

1. Do EDA to select relevant features
2. Perform clustering (Kmeans, GMM, DBScan, Agglomeartive, Spectral), to see if the selected features separate the data and also gauge the shape and density of the points in feature space to select appropriate classifier
3. If we're able to separate the points as we expect to we'd like to implement any of the probablity based classifier such as decision tree or Naive Bayes

Question 5: Before selecting the methods you listed in response to Question 4, what are all methods you originally considered to use for the selected data mining task? What was your rationale for selecting the methods you listed in response to Question 4? What was your rationale for not selecting other methods?

Originally the idea was to do trail and error for each classifier learned in the course with all of the attributes, but after careful considerations of many other factors (such as the learning time for SVM algoritms and distortion due to noise), we decided we'd be better off if we correctly guess the data in feature space and then apply appropriate classifier on it.

Question 6: What limitations does your 'selected' method(s) has(have) that may limit your ability to accomplish the goal you have set for yourself?

There's a very good chance that our classes are not well defined (good separation and good cohesiveness) and clustering doesn't reveal anything about the shape or the density of the data

Question 7: Do you have any alternative plan/strategy to overcome the above limitation(s)?

if nothing at all is revealed then in worst case we'd do a trial and error process to find the best classifier

Question 8: For each of the methods you want to use, what parameter choices do you want to use and why? It does not have to be one parameter choice, it could be a collection or a range of choices you may want to consider.

Clustering: Ideally We'd like to find 2 clusters (with no noise for DB Scan) for each of the algorithms
Classfication: For Decision Tree we expect the max nodes to be in range of 3 to 8, we're expecting the selected space to be much lower than the original 21 dimensional space, anything more than 8 we think would overfit the data and anything less than 3 would cause underfitting. we don't have to worry about parameters for Naive Bayes.

Question 9: How will you evaluate that you are successful in your pursuing your goal at the end of the project? In other words, what is your evaluation criteria?

There're 3 part to the goal as initially mentioned

1. EDA: We'd consider ourselves successful if we're able to give atleast give 3 direct recommendations (i.e find some interesting relationships between the attributes) to the management based on the EDA
2. Clustering: Deduce most suitable classifier based clustering analysis done on the data
3. Classification: Considering the low training size (36 employees in total)Ideally we'd like our model to predict the probablity of an emplloyee being High Absentee atleast 2/3rd of the times correctly.

Question 10: How will you evaluate that you are successful in your pursuing your goal at the end of the project? In other words, what is your evaluation criteria?

There're 2 part to the goal as initially mentioned

1. We'd consider ourselves successful if we're able to give atleast give 3 direct recommendations to the management based on the EDA.
2. Classification: Considering the low training size (36 employees in total)Ideally we'd like our model to predict the probablity of an emplloyee being High Absentee atleast 2/3rd of the times correctly.

Question 11: Show any visualizations you may have generated to understand your data. Please include the code you used and the plots below. If you borrowed code (entirely or partially) from the hands-on projects or anywhere else, clearly provide a link to your source.

You may use this package to load UCI data in python: https://github.com/SkafteNicki/py_uci (https://github.com/SkafteNicki/py_uci)

1. EDA and Feature Selection

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import seaborn as sns
```

In [74]:
```
# Data Load
data = pd.read_excel('C:/Users/nifaullah/Downloads/Absenteeism_at_work_AAA/Abs
enteeism_at_work.xls')
# Transforming dataset from each absent record to user specific summary
testz = data.groupby(['ID'])[['Absenteeism time in hours']].sum()
testz['id'] = testz.index.values
high_absentees = testz[testz['Absenteeism time in hours'] > 200].iloc[:,1]
high_absentees_df = data[data['ID'].isin(high_absentees)]
low_absentees_df = data[~data['ID'].isin(high_absentees)]
# Removal of Columns which cannot be logically summarised
high_absentees_df_demo = high_absentees_df.drop(['Month of absence','Day of th
e week','Seasons', 'Reason for absence'], axis = 1)
low_absentees_df_demo = low_absentees_df.drop(['Month of absence','Day of the
 week','Seasons', 'Reason for absence'], axis = 1)
# Creating user specific summary dataset
high_absentees_df_demo1 = high_absentees_df_demo.groupby('ID').agg(
    {'ID': 'mean',
     'Transportation expense': 'mean',
     'Distance from Residence to Work': 'mean',
     'Service time': 'mean',
     'Age': 'mean',
     'Hit target': 'mean',
     'Disciplinary failure': 'mean',
     'Son': 'mean',
     'Social drinker': 'mean',
     'Social smoker': 'mean',
     'Pet': 'mean',
     'Education': 'mean',
     'Body mass index': 'mean',
     'Absenteeism time in hours': ['mean', 'sum']
    })
low_absentees_df_demo1 = low_absentees_df_demo.groupby('ID').agg(
    {'ID': 'mean',
     'Transportation expense': 'mean',
     'Distance from Residence to Work': 'mean',
     'Service time': 'mean',
     'Age': 'mean',
     'Hit target': 'mean',
     'Disciplinary failure': 'mean',
     'Son': 'mean',
     'Social drinker': 'mean',
     'Social smoker': 'mean',
     'Pet': 'mean',
     'Education': 'mean',
     'Body mass index': 'mean',
     'Absenteeism time in hours': 'mean',
     'Absenteeism time in hours': ['mean', 'sum']
    })
# Concatenate users
result = pd.concat([high_absentees_df_demo1, low_absentees_df_demo1])
# Create Class Column to specify class of each users based on Absenteeism time
in hours
result['class'] = result['Absenteeism time in hours']['sum'].apply(lambda x: 0
if x < 200
                                                                            els
e 1)
```

```
result.columns = ['ID', 'Transportation expense', 'Distance from Residence to
 Work', 'Service time', 'Age', 'Hit target',
                  'Disciplinary failure', 'Son', 'Social drinker', 'Social smo
ker', 'Pet', 'Education', 'Body mass index',
                  'Absenteeism time in hours Mean', 'Absenteeism time in hours
Sum', 'class']
```

In [259]:
```
result.head()
```

Out[259]:

ID	ID	Transportation expense	Distance from Residence to Work	Service time	Age	Hit target	Disciplinary failure	Son	Social drinker	Social smoker
3	3	179	51.0	18.0	38.0	95.070796	0.008850	0.0	1.0	0
9	9	228	14.0	16.0	58.0	95.875000	0.000000	2.0	0.0	0
11	11	289	36.0	13.0	33.0	93.850000	0.050000	2.0	1.0	0
14	14	155	12.0	14.0	34.0	95.689655	0.000000	2.0	1.0	0
15	15	291	31.0	12.0	40.0	92.594595	0.054054	1.0	1.0	0

In [260]:
```
result.tail()
```

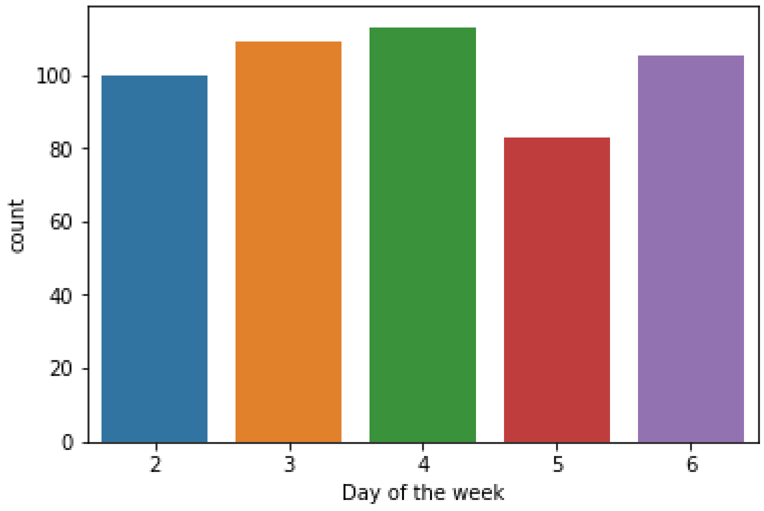
Out[260]:

ID	ID	Transportation expense	Distance from Residence to Work	Service time	Age	Hit target	Disciplinary failure	Son	Social drinker	Social smoker
30	30	157	27.0	6.0	29.0	93.714286	0.142857	0.0	1.0	1
31	31	388	15.0	9.0	50.0	96.666667	0.333333	0.0	0.0	0
32	32	289	48.0	29.0	49.0	92.600000	0.000000	0.0	0.0	0
33	33	248	25.0	14.0	47.0	94.791667	0.041667	2.0	0.0	0
35	35	179	45.0	14.0	53.0	95.000000	0.000000	1.0	0.0	0

First let's see if we there is any particular day of the week (like weekends) where people are absent more

In [111]:
```
sns.countplot(x ='Day of the week', data = data[data['ID'].isin(high_absentees
)])
```

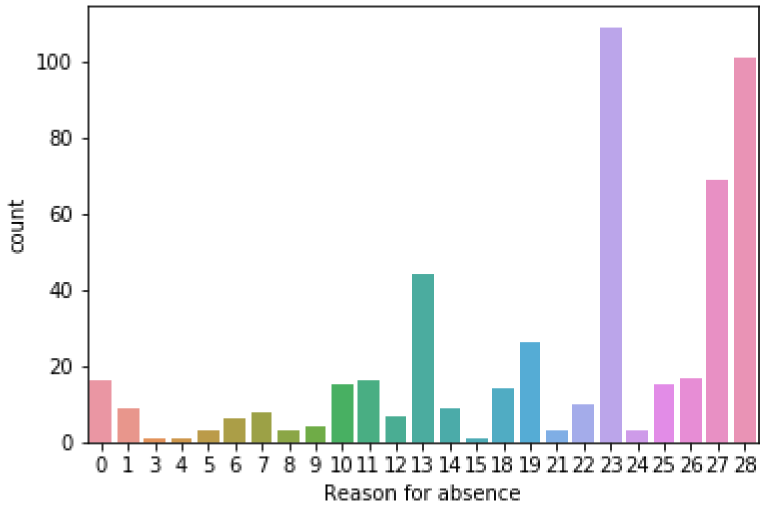
Out[111]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bbc64518>
```



Turns out there isn't much big of a difference, but people are less absent on Thursday are slightly less, so may be management if it plans to conduct any important activity for entire employees it can conduct on Thursday

In [72]:
```
sns.countplot(x ='Reason for absence', data = data[data['ID'].isin(high_absent
ees)])
```

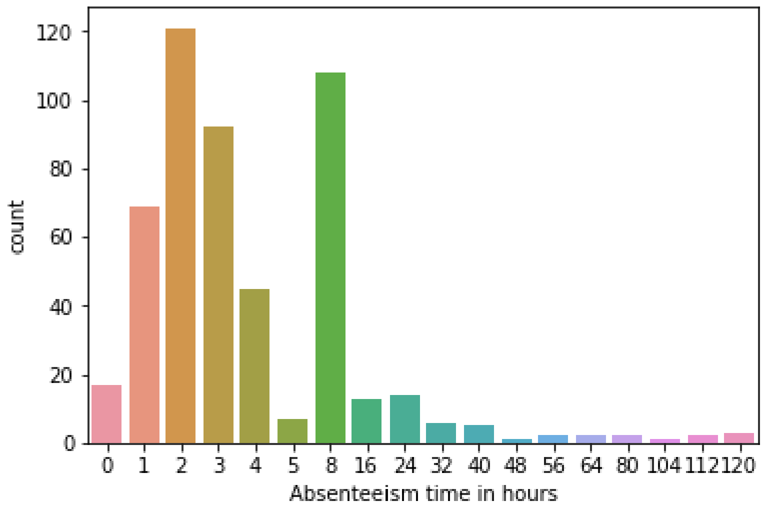
Out[72]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202ba56bba8>
```



Medical consultation and Dental consultation seems to be most popular reason People with high absenteeism

```
In [73]: sns.countplot(x ='Absenteeism time in hours', data = data[data['ID'].isin(high
         _absentees)])
```

```
Out[73]: <matplotlib.axes._subplots.AxesSubplot at 0x202ba624f60>
```

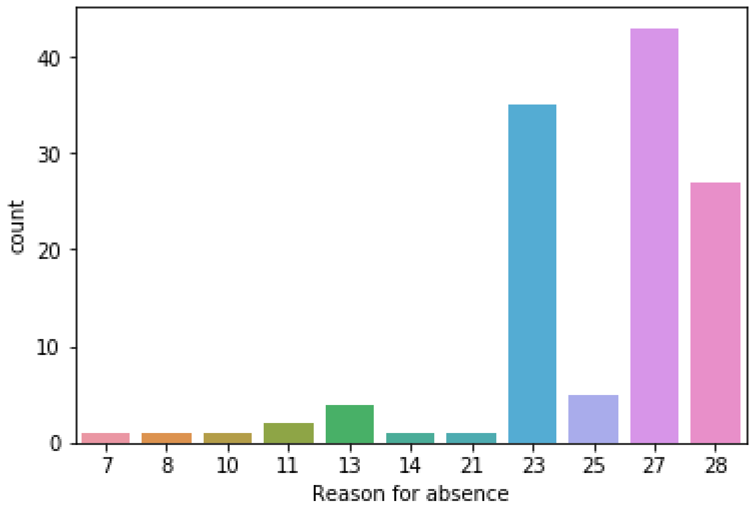


People with high absenteeism mostly seem to be absent either mostly for 2 hours or 8 hours. Now let's see what're the most popular reasons for 2 hr absentation. Also there're small chunks of employees who have long continuos absentation suggesting some extended, we'd also like to know most common reason for such long absentation.

```
In [84]: high_abs = data[data['ID'].isin(high_absentees)]
         low_abs = data[~data['ID'].isin(high_absentees)]
```

```
In [83]: sns.countplot(x ='Reason for absence', data = high_abs[high_abs['Absenteeism t
         ime in hours'] == 2])
```

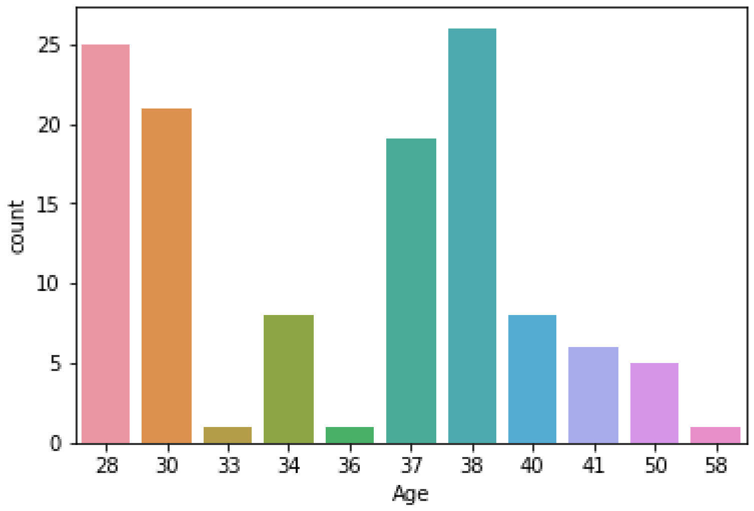
```
Out[83]: <matplotlib.axes._subplots.AxesSubplot at 0x202ba7b9c18>
```



A large number of people are taking 2 hour absentation for physiotherapy, medical consultation and dental consultation, which seems absurd. One course of action for the management may be to put up monthly dental or medical camps, this would most likely cut down the absentation by huge amount.

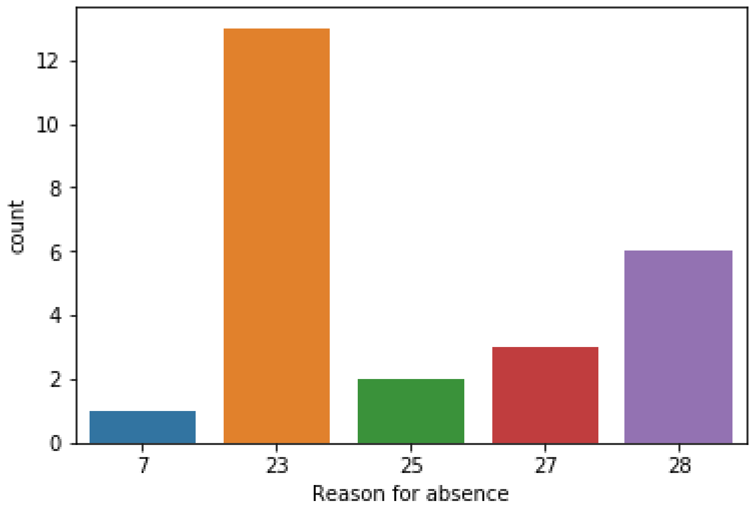
In [87]:
```
sns.countplot(x ='Age', data = high_abs[high_abs['Absenteeism time in hours']
== 2])
```

Out[87]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202ba909c88>
```



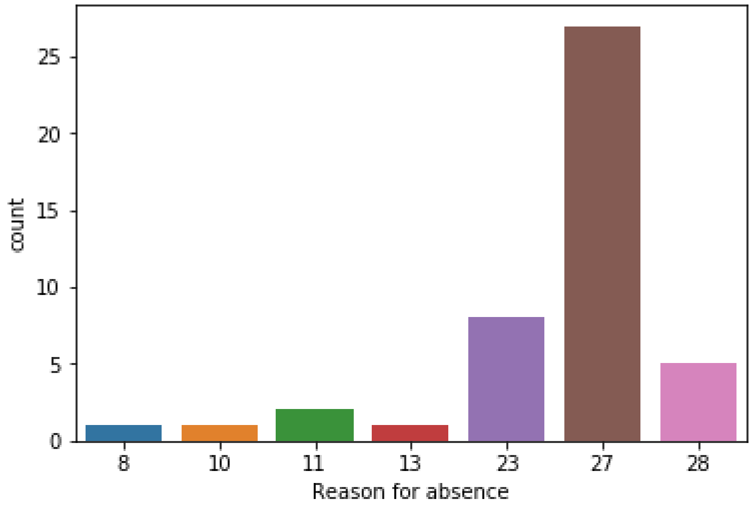
In [91]:
```
sns.countplot(x ='Reason for absence', data = high_abs[(high_abs['Absenteeism
 time in hours'] == 2) & (high_abs['Age'] == 28)])
```

Out[91]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202ba980898>
```



In [96]:
```
sns.countplot(x ='Reason for absence', data = high_abs[(high_abs['Absenteeism
 time in hours'] == 2) & (high_abs['Age'].isin([37,38]))])
```

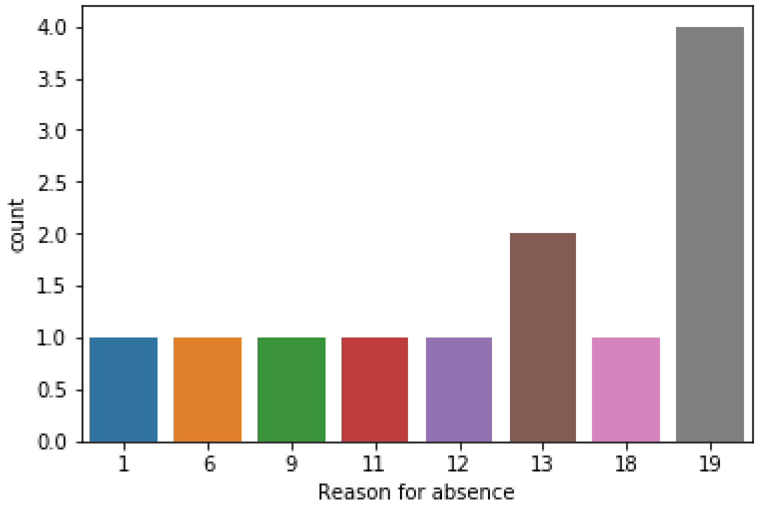
Out[96]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bab5d828>
```



Again there seems to be a very clear pattern here 28 year old employees are often taking 2 hours off for medical consultation and 37-38 year olds are taking off for Dental consultation, which again is very surprising normally you'd expect it to be other way around.

In [257]:
```
sns.countplot(x ='Reason for absence', data = high_abs[high_abs['Absenteeism t
ime in hours'] >= 50])
```

Out[257]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202e307deb8>
```

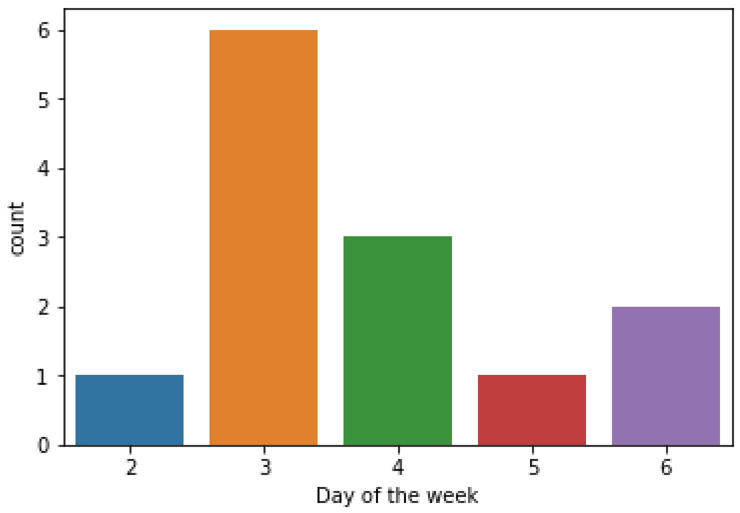


Nothing suspicious here, people taking long leaves for Injury, poisoning and certain other consequences of external causes seems perfectly reasonable

In [94]:
```
sns.countplot(x ='Day of the week', data = high_abs[(high_abs['Absenteeism tim
e in hours'] == 2) & (high_abs['Age'] == 28)
                                                    & (high_abs['Reason for ab

sence'] == 23)])
```

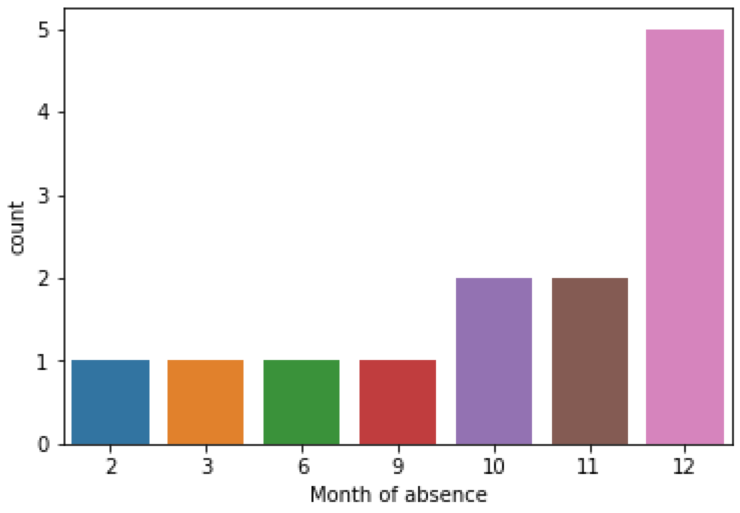
Out[94]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202baa9d358>
```



In [95]:
```
sns.countplot(x ='Month of absence', data = high_abs[(high_abs['Absenteeism ti
me in hours'] == 2) & (high_abs['Age'] == 28)
                                                    & (high_abs['Reason for ab

sence'] == 23)])
```

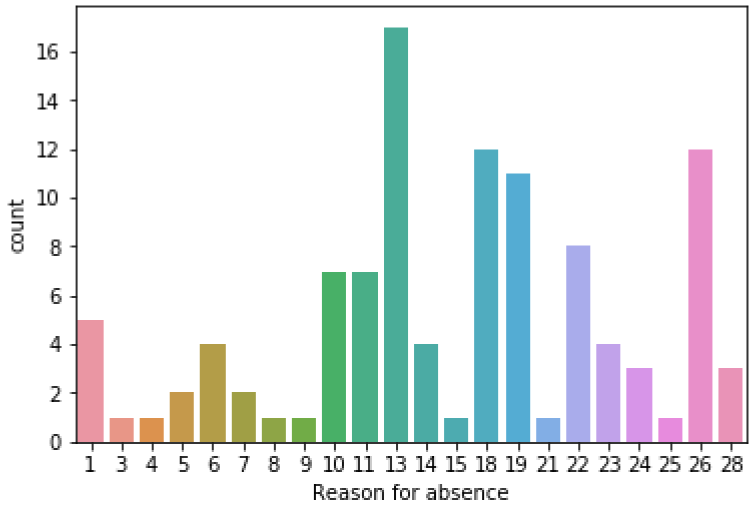
Out[95]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bab002e8>
```



Day of the week and Month of absence doesn't reveal any suspicious pattern

```
In [85]: sns.countplot(x ='Reason for absence', data = high_abs[high_abs['Absenteeism t
         ime in hours'] == 8])
```

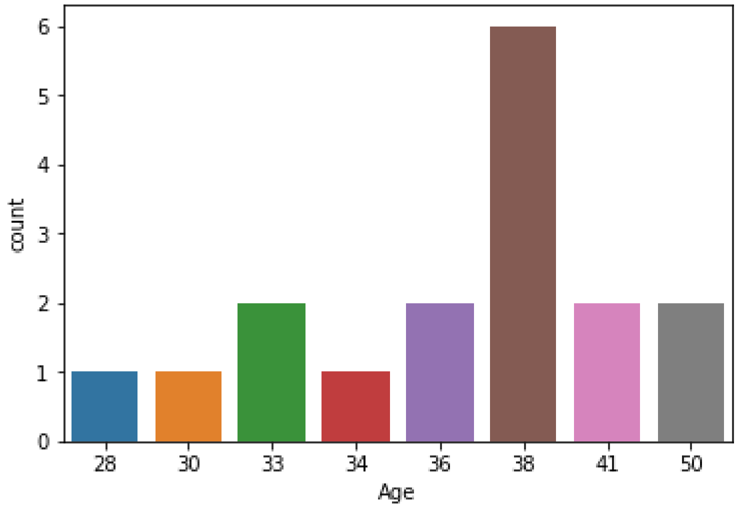
```
Out[85]: <matplotlib.axes._subplots.AxesSubplot at 0x202ba6e8a20>
```



People with muscoskeletal problems seems to be most with 8hrs absentations which seems to be genuine.

```
In [101]: sns.countplot(x ='Age', data = high_abs[(high_abs['Absenteeism time in hours']
          == 8)
                                                   & (high_abs['Reason for absence'] == 1
          3)])
```

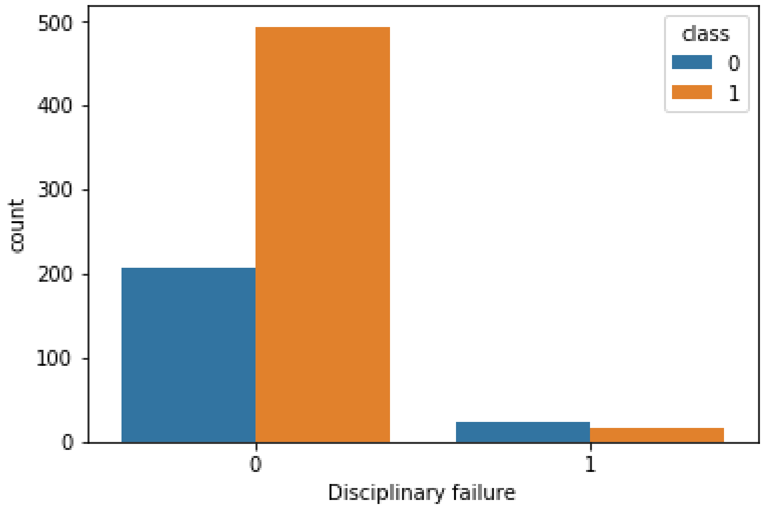
```
Out[101]: <matplotlib.axes._subplots.AxesSubplot at 0x202bab4ba58>
```



Nothing out of place is here

```
In [107]: full_df = pd.concat([high_abs, low_abs])
```

In [110]:
```
sns.countplot(x = 'Disciplinary failure', hue = 'class', data = full_df)
```
Out[110]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bbc0d048>
```



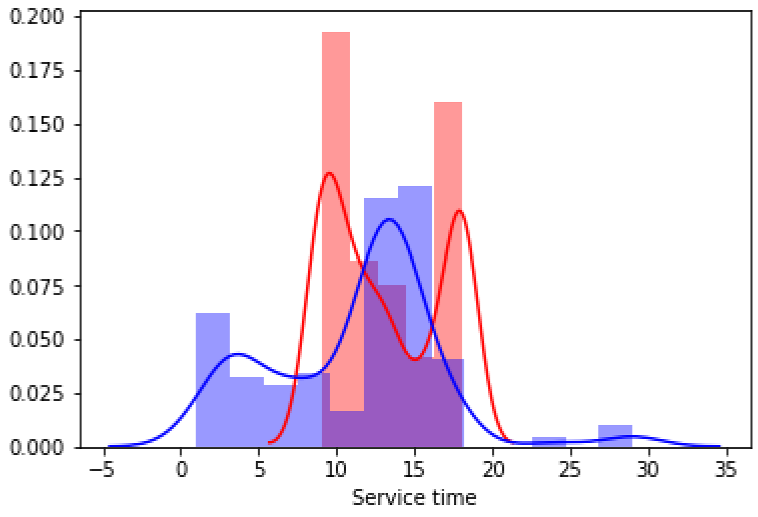
People with low absenteeism rate have proportionately higher representative in disciplinary failures, this could suggest that people who have been dealt with disciplinary action tend to be absent less.

Again another course of action for the management here can be to proceed with disciplinary action for any excessive absenteesim, evidence from the above graph suggest this would most likely aid in reducing the absenteeism

In [121]:
```
high_abs.columns
```
Out[121]:
```
Index(['ID', 'Reason for absence', 'Month of absence', 'Day of the week',
       'Seasons', 'Transportation expense', 'Distance from Residence to Wor
k',
       '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', 'class'],
      dtype='object')
```

In [124]:
```
sns.distplot(high_abs['Service time'], color = 'red')
sns.distplot(low_abs['Service time'], color = 'blue')
```

Out[124]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bbde9e10>
```



This again fits well with our earlier age analysis, employees with 8 - 12 years of service time display high absenteeism, most employee with service time betweeen 13 -17 years show low absenteeism and again employees with 18 - 21 years of service time are showimg igh absenteeism.

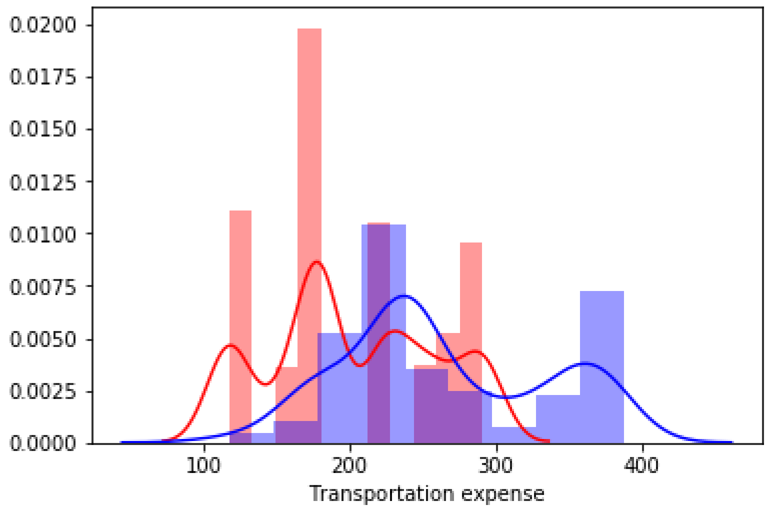
This is a very strange cluster with people different belonging age groups on either side of a thick region behave similarly.Due to the difference in their service time and ages we can rule out common cause.

Employees with 8 -12 years of experience generally have a very young family to deal with (again we cannot comment more on this because of lack of data) but this could be one of the hypothesis. Management can take steps such as installation of a Day care center or moving the office near to the closest school district might help reducing the absenteeism problem for this group.

Employees with 18 - 21 years of experience generally are very hard to motivate, inaddtion to the expected health issues. Management can run programs to motivate such employees, thus helping the in the curbing of absenteeism

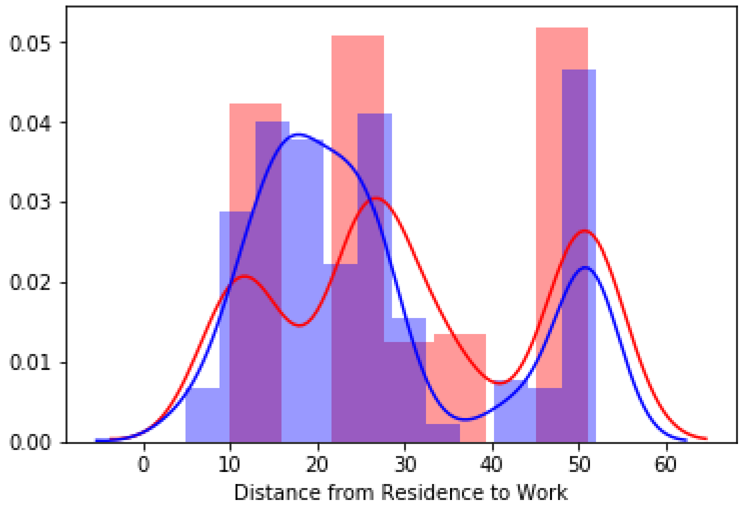
In [125]:
```
sns.distplot(high_abs['Transportation expense'], color = 'Red')
sns.distplot(low_abs['Transportation expense'], color = 'Blue')
```

Out[125]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bbea4cc0>
```



In [126]:
```
sns.distplot(high_abs['Distance from Residence to Work'], color = 'Red')
sns.distplot(low_abs['Distance from Residence to Work'], color = 'Blue')
```

Out[126]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bbe9a550>
```

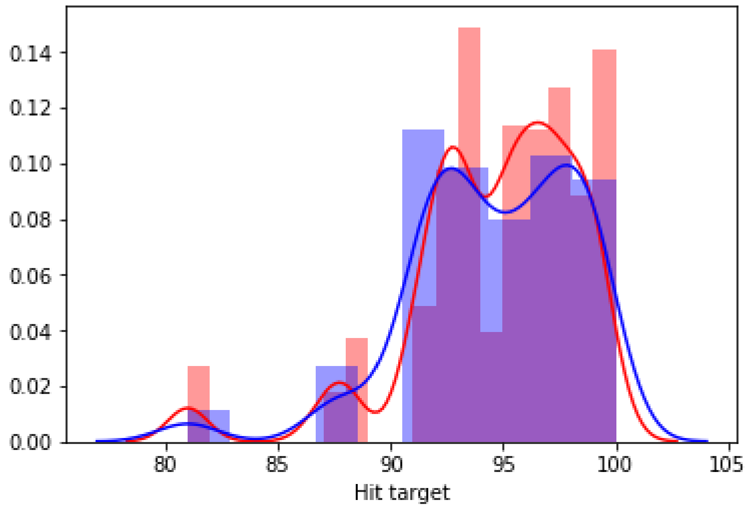


People who have less transportation expenses seem to be more absent but yet distance from residence to work doesn't show the same trend, suggesting that these employees may well be using public transport service to reach office which is often causing the delay or excessive absenteeism of 2 hours.

We need further analysis on this, if the above hypothesis is provn to be true, perhaps the management can run it's own transportation service to reduce absenteeism

In [128]:
```
sns.distplot(high_abs['Hit target'], color = 'Red')
sns.distplot(low_abs['Hit target'], color = 'Blue')
```

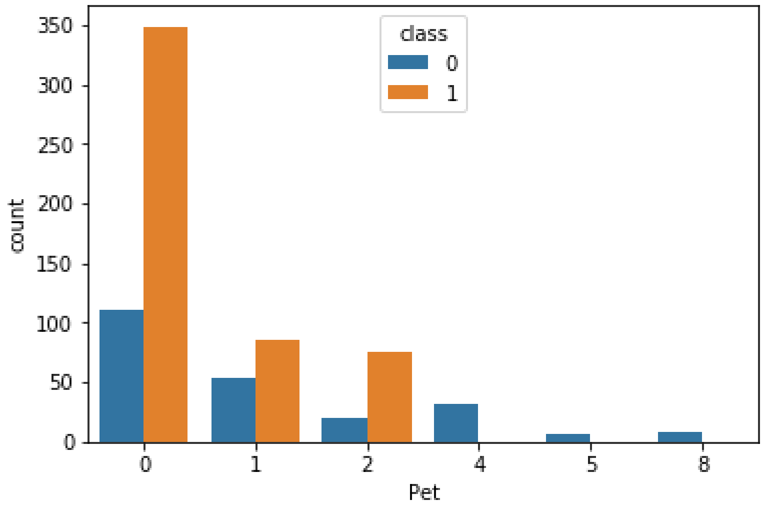
Out[128]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bbf89ac8>
```



Nothing of note here

In [132]:
```
sns.countplot(x='Pet', hue = 'class', data = full_df)
```

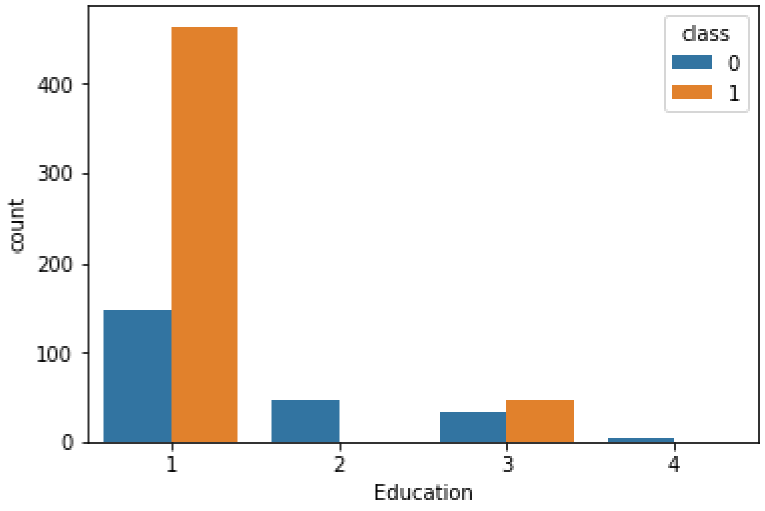
Out[132]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bc048668>
```



Ineteresting with no pet are more prone to absenteeism, we need to investigate more before we claim anything

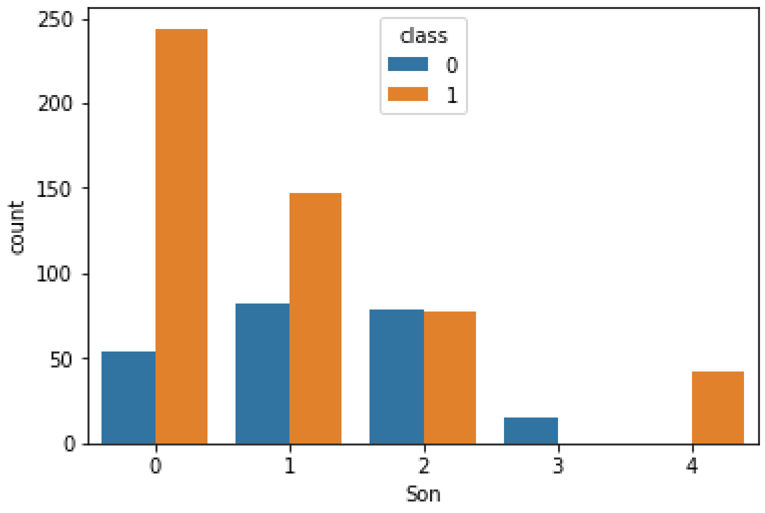
In [134]:
```
sns.countplot(x='Education', hue = 'class', data = full_df)
```

Out[134]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bc0b3a58>
```



In [136]:
```
sns.countplot(x='Son', hue = 'class', data = full_df)
```

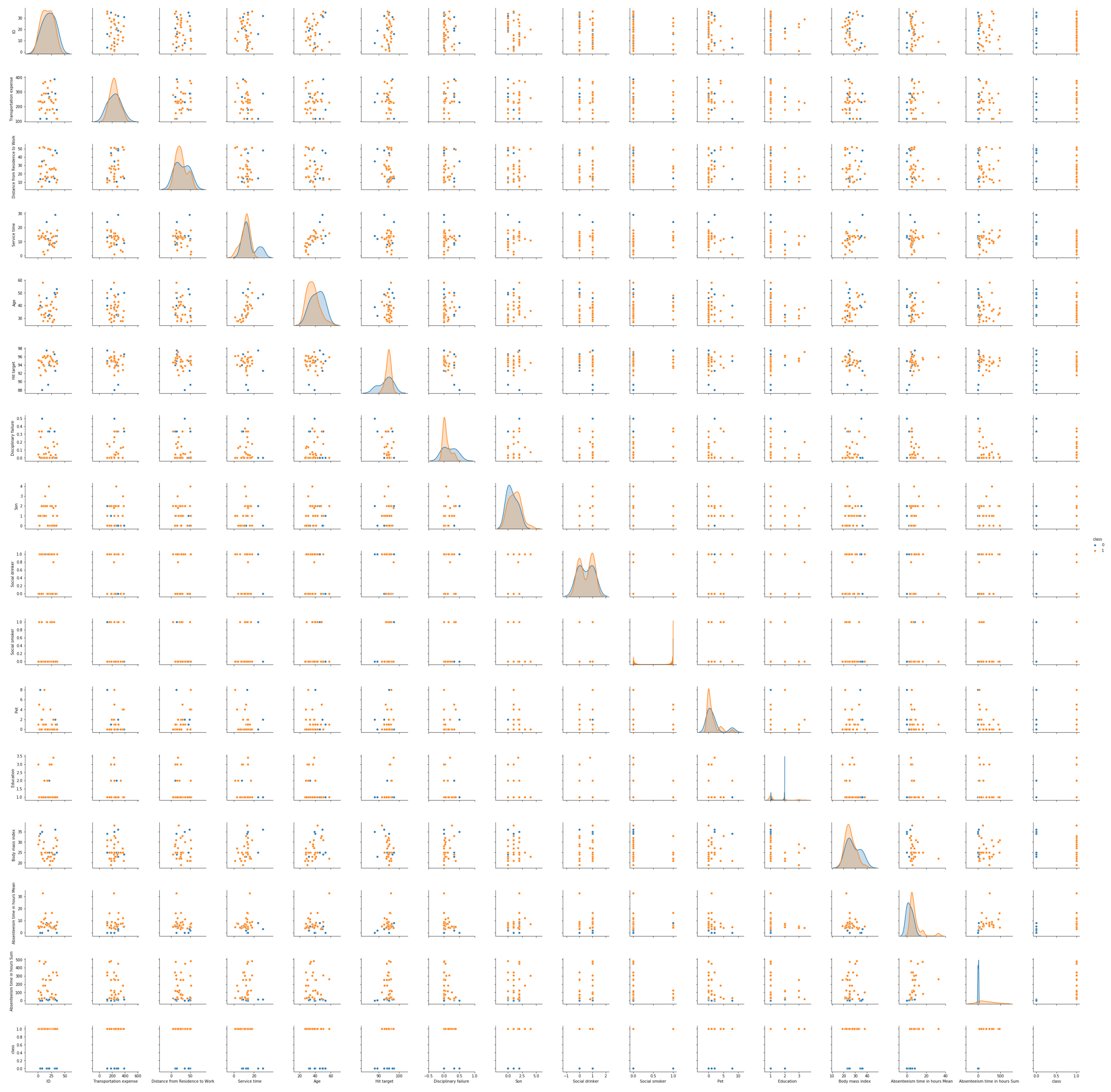
Out[136]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202bc13a518>
```



Proportionately, both eductaion and number of children doesn't seem to raise any markers

In [141]:
```
sns.pairplot(result, hue='class')
```

Out[141]:
```
<seaborn.axisgrid.PairGrid at 0x202d307f8d0>
```



Based on the above pairplots (more specfically the diagonal class wise distribution and the last column) and class wise univariate analysis (see plots and inferences above the pairplots) we narrow down to these features 'Transportation expense', 'Service time', 'Disciplinary failure', 'Pet'. We believe below features would help significantly to segregate the classes

In [138]:
```
features = ['Transportation expense','Service time', 'Disciplinary failure',
'Pet', 'class']
```

Question 12: **Perform data mining, evaluate your work and report your findings.** This should include code, plots and results you may have generated. If you borrowed code (entirely or partially) from the hands-on projects or anywhere else, clearly provide a link to your source.

2. Data Analysis using different Clustering Algorithms

In [219]:
```
from sklearn.mixture import GaussianMixture
from sklearn.cluster import AgglomerativeClustering
from sklearn.cluster import DBSCAN
from sklearn.cluster import SpectralClustering
from sklearn.cluster import KMeans

from scipy.special import comb
def rand_index(S, T):

    Spairs = comb(np.bincount(S), 2).sum()
    Tpairs = comb(np.bincount(T), 2).sum()

    A = np.c_[(S, T)]

    f_11 = sum(comb(np.bincount(A[A[:, 0] == i, 1]), 2).sum()
             for i in set(S))

    f_10 = Spairs - f_11
    f_01 = Tpairs - f_11
    f_00 = comb(len(A), 2) - f_11 - f_10 - f_01
    return (f_00 + f_11) / (f_00 + f_01 + f_10 + f_11)
```

In [185]:
```
result['class'] = result['Absenteeism time in hours Sum'].apply(lambda x: 0 if
x <= 200
                                                                            els
e 1)
```

In [186]:
```
full_demo_df = result[features]
```

In [187]:
```
test_demo_df = full_demo_df.iloc[:,0:4]
```

In [188]:
```
test_demo_df.columns
```

Out[188]:
```
Index(['Service time', 'Disciplinary failure', 'Transportation expense',
       'Pet'],
      dtype='object')
```

In [189]:
```
test_demo_df.shape
```

Out[189]:
```
(36, 4)
```

In [224]:
```
kmeans = KMeans(n_clusters=2, random_state=10);
y_pred = kmeans.fit_predict(test_demo_df)
print(rand_index(y_pred, full_demo_df.iloc[:,-1]))
```

```
0.4857142857142857
```

Nearly half of the points are clustered correctly. This only seems to suggest that the clusters are not naturally well separated.

```
In [225]: gmm = GaussianMixture(n_components=2, covariance_type='full')
          y_pred = gmm.fit_predict(test_demo_df)
          print(rand_index(y_pred, full_demo_df.iloc[:,-1]))

          0.5
```

Again suugesting that points are not well separated, the overlapping of data from different clusters messed up the probabilities to each distribution

```
In [211]: dbscan = DBSCAN(eps=37, min_samples=5)
          y_pred = dbscan.fit_predict(test_demo_df)
          sum(y_pred == -1)

Out[211]: 4

In [212]: np.unique(y_pred)

Out[212]: array([-1,  0,  1], dtype=int64)

In [226]: print(rand_index(y_pred, full_demo_df.iloc[:,-1]))

          0.5
```

The above results suggests that points within clusters are loosely coupled and scattered around

```
In [227]: single_linkage = AgglomerativeClustering(linkage="single", n_clusters=2)
          y_pred = single_linkage.fit_predict(test_demo_df)
          print(rand_index(y_pred, full_demo_df.iloc[:,-1]))

          0.5634920634920635

In [228]: complete_linkage = AgglomerativeClustering(linkage="complete", n_clusters=2)
          y_pred = complete_linkage.fit_predict(test_demo_df)
          print(rand_index(y_pred, full_demo_df.iloc[:,-1]))

          0.5253968253968254

In [229]: average_linkage = AgglomerativeClustering(linkage="average", n_clusters=2)
          y_pred = average_linkage.fit_predict(test_demo_df)
          print(rand_index(y_pred, full_demo_df.iloc[:,-1]))

          0.49206349206349204
```

In [230]:
```
n_clusters = 2
spectral = SpectralClustering(n_clusters=n_clusters, random_state=10)
y_pred = spectral.fit_predict(test_demo_df)
print(rand_index(y_pred, full_demo_df.iloc[:,-1]))
```

```
0.5428571428571428

C:\Users\nifaullah\AppData\Roaming\Python\Python37\site-packages\sklearn\mani
fold\spectral_embedding_.py:235: UserWarning: Graph is not fully connected, s
pectral embedding may not work as expected.
  warnings.warn("Graph is not fully connected, spectral embedding"
```

Again reiterating the same points as above that the points intra cluster are not cohesive and inter cluster are not well separated

The above cluster analysis has made it really difficult for us to make a straight forward choice because we assumed we'd get a well separated space, but yet based on the features selected and derived feature space and shape from clustering analysis, we believe that decision tree because of it's axis paralell partioning system should do a reasonable job in comparison to other classifiers since we selected features such that it'd differentiate the classes. We believe Naive Bayes as opposed to assumed earlier is likely not to do as well as Decision Tree because data is scattered and classes most likely overlap

3. Classification: Test the conclusions of above cluster analysis

In [231]:
```
#Classification Algorithms
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
```

In [232]:
```
class_demo_df = full_demo_df.iloc[:,-1]
class_demo_df.shape
```

Out[232]:
```
(36,)
```

In [233]:
```
dt2 = DecisionTreeClassifier(max_depth=2)
dt4 = DecisionTreeClassifier(max_depth=4)
dt6 = DecisionTreeClassifier(max_depth=6)
dt8 = DecisionTreeClassifier(max_depth=8)
dt3 = DecisionTreeClassifier(max_depth=3)
dt5 = DecisionTreeClassifier(max_depth=5)
```

In [239]:
```
dt2_score = cross_val_score(dt2, test_demo_df, class_demo_df, cv=10, scoring=
'accuracy')
dt4_score = cross_val_score(dt4, test_demo_df, class_demo_df, cv=10, scoring=
'accuracy')
dt6_score = cross_val_score(dt6, test_demo_df, class_demo_df, cv=10, scoring=
'accuracy')
dt8_score = cross_val_score(dt8, test_demo_df, class_demo_df, cv=10, scoring=
'accuracy')
dt3_score = cross_val_score(dt3, test_demo_df, class_demo_df, cv=10, scoring=
'accuracy')
dt5_score = cross_val_score(dt5, test_demo_df, class_demo_df, cv=10, scoring=
'accuracy')
```

In [240]:
```
dt_accuracy = { 'Depth': [2,4,6,8,3,5],
    'Score': [dt2_score.mean(),dt4_score.mean(),dt6_score.mean(),dt8_score.mea
n(), dt3_score.mean(),
                          dt5_score.mean()]
                   }
dt_accuracy_df = pd.DataFrame.from_dict(dt_accuracy)
print(dt_accuracy_df)
```

```
   Depth     Score
0      2  0.565000
1      4  0.690000
2      6  0.693333
3      8  0.685000
4      3  0.665000
5      5  0.685000
```

69% accuracy for 6 rules seems very reasonable for such a small sample (36 employees) and also 6 rules for 4 attributes is also very reasonable.

In [242]:
```
nb = GaussianNB()
nb_scores_1 = cross_val_score(nb, test_demo_df, class_demo_df, cv=10, scoring=
'accuracy')
nb_scores_1.mean()
```

Out[242]:
```
0.6716666666666666
```

Surprisingly Naive Bayes also does a very reasonable job, considering that we were failing to naturally cluster the space. May be the fact that it assumes that probabability of each attribute to be independent of each other actually helps it to genarlize the feature space and saves from overfitting

Although our goal has been met, we'd like to test few other classifiers to see if our inference about the data being decision tree friendly because of the feature space and the results of cluster analysis, is true or not

In [244]:
```
knn1 = KNeighborsClassifier(n_neighbors=1)
knn3 = KNeighborsClassifier(n_neighbors=3)
knn5 = KNeighborsClassifier(n_neighbors=5)
knn7 = KNeighborsClassifier(n_neighbors=7)
knn1_score = cross_val_score(knn1, test_demo_df, class_demo_df, cv=10, scoring
='accuracy')
knn3_score = cross_val_score(knn3, test_demo_df, class_demo_df, cv=10, scoring
='accuracy')
knn5_score = cross_val_score(knn5, test_demo_df, class_demo_df, cv=10, scoring
='accuracy')
knn7_score = cross_val_score(knn7, test_demo_df, class_demo_df, cv=10, scoring
='accuracy')
knn_accuracy_df = pd.DataFrame.from_dict({ 'Neighbours': [1,3,5,7],
    'Score': [knn1_score.mean(),knn3_score.mean(),knn5_score.mean(),knn7_score
.mean()]})
print(knn_accuracy_df)
```

```
   Neighbours     Score
0           1  0.465000
1           3  0.431667
2           5  0.590000
3           7  0.648333
```

Knn doesn't perform as good as decision tree or naive bayes but it's performance improves with generalization. Again this is most likely to suffer because as we know classes are overlapping and scattered.

In [245]:
```
svm_linear = SVC(C=0.1, kernel='linear')
svm_scores_1 = cross_val_score(svm_linear, test_demo_df, class_demo_df, cv=10,
scoring='accuracy')
svm_scores_1.mean()
```

Out[245]:
```
0.6100000000000001
```

In [246]:
```
svm_linear = SVC(C=10, kernel='linear')
svm_scores_1 = cross_val_score(svm_linear, test_demo_df, class_demo_df, cv=10,
scoring='accuracy')
svm_scores_1.mean()
```

Out[246]:
```
0.6100000000000001
```

SVM Linear also as expected (because of the way how clustering panned out) fails to do as a good job as Decision Tree

In [250]:
```
svm_rbf_001 = SVC(C = 0.5, kernel='rbf', gamma=0.01)
svm_rbf_01 = SVC(C = 0.5, kernel='rbf', gamma=0.1)
svm_rbf_1 = SVC(C = 0.5, kernel='rbf', gamma=1)
svm_rbf_scores_001 = cross_val_score(svm_rbf_001, test_demo_df, class_demo_df, cv=10, scoring='accuracy')
svm_rbf_scores_01 = cross_val_score(svm_rbf_01, test_demo_df, class_demo_df, cv=10, scoring='accuracy')
svm_rbf_scores_1 = cross_val_score(svm_rbf_1, test_demo_df, class_demo_df, cv=10, scoring='accuracy')
svm_rbf_accuracy_df = pd.DataFrame.from_dict({ 'Order': ['001','01','1'],
    'Score': [svm_rbf_scores_001.mean(),svm_rbf_scores_01.mean(),svm_rbf_scores_1.mean()]})
print(svm_rbf_accuracy_df)
```

```
  Order     Score
0   001  0.693333
1    01  0.693333
2     1  0.693333
```

SVM RBF manages to match the accuracy of the decision tree, which tells us even after projecting the points into a higher dimensional space doesn't really improve upon the simple decision tree classifier

In [253]:
```
svm_poly_scores = cross_val_score(SVC(C=0.01, kernel='poly',degree=1, gamma = 'auto'), test_demo_df, class_demo_df, cv=10, scoring='accuracy')
svm_poly_scores.mean()
```

Out[253]:
```
0.6683333333333333
```

In [254]:
```
svm_poly_scores = cross_val_score(SVC(C=0.01, kernel='poly',degree=2, gamma = 'auto'), test_demo_df, class_demo_df, cv=10, scoring='accuracy')
svm_poly_scores.mean()
```

Out[254]:
```
0.4766666666666667
```

In [255]:
```
svm_poly_scores = cross_val_score(SVC(C=0.01, kernel='poly',degree=3, gamma = 'auto'), test_demo_df, class_demo_df, cv=10, scoring='accuracy')
svm_poly_scores.mean()
```

Out[255]:
```
0.45166666666666666
```

SVM Polynomial with degrees 1, 2 and 3 fail to improve the score of the decision tree. And additionally when we project the points in high dimensions accuracy decreases more, suggesting that similar points are not grouping as expected in higher dimensional space

4. Conclusion

In the end our hypothesis that decision tree is the best classifer, based on the cluster analysis seems very likely to be true.

We'd say that our approach, to understand the nature of the data in feature space with the application of clustering and classfication knowledge we posseses, has been reasonably successful.

5. References

Most code used here are from IDA Clustering and Classification Hands On excercise and practise. Accuracy measured are based on the reliable cross validation methodology

Question 13: Putting your findings in the context of your goal and evaluation plan, do you consider yourself successful? Provide reasons for your success or lack thereof.

Yes we consider ourselves to be substantially succesful if not entirely successful. When we started with our analysis we had three 3 specific objectives, which we believe has been substantially fulfilled.

Goal 1 - Come up with attributes which differntiate or identify attributes separating High absentees from low absentees. Based on these attributes provide atleast 3 course of actions for the management
Result 1 - Using extensive EDA we were not only able reduce the feature space but also able to provide 4 concrete ation items and 2 not so concrete but good enough to be pondered action items for the management to work upon.

Goal 2 - Choose the best classsifier based upon the clustering of the points in feature space.
Result 2 - Although clustering analysis didn't help the way we expected it to help (i.e we expected the data to be have good intra-classes cohesion and high inter-classes separation) but because of the results we were able to rule out most of the classifiers and selection of feature space that differntiate the classes pushed us towards the decision tree classifier as it draws axis (feature) parallel boundaries. And Decison Tree turned out to be the best performing and least complex algorithm, which points to the success of the process. We then run a extensive analysis of other classifiers to prove that indeed decision tree classifier is the best classifier for the selected feature space

Goal 3: Develop a model with atleast 2/3rd of accuracy to predict if a person is likely to be High Absentee.
Result 3: We successfully developed a decision tree model with 6 nodes with 69% accuracy, which is reasonably good considering the low sample size (36 unique employees).

Question 14: If you have an extra month to work on this project, what else would you do? Provide reasons.

1. First of all we would like to find more relation between the attributes. for example I would want to find difference in reason provided for large absentation (say 30 hours or more) incomparison with relatively small absentations (say 2 hours).
2. Then we would like to look into the socio-economic, health and sanitation factors of the said area to see if they affect the high absentation rate
3. We would like to build a ratings based classifier (kind of decision tree) for a model instead of hard classifier we have now which would judge a person on all four features and then provide a rating based on those features and the management would then decide based on that. For example a person with rating of 4 would have a very high chance of High absenteeism and a person with rating 3 will have slightly less and so on. Additionally we'd also like to try out few other features which can to the model to improve upon it.
4. We would also like to run a SVD anlysis, if succesful we'd have a very good idea about the prototypical high absentation employee and major atttributes of high absenteeism.
5. We would also like to run our model for different type of companies around that area and also to courier companies not belonging to the area, to draw more factors affecting the high absenteeism.
6. Also, if looked closely this seems to be more than 2 class problem because of the way data is distributed, we'd like to do some analysis on this too.
7. on similar lines as above We are also not entirely sure of the class separation criteria (i.e anybody above 199 hours absentation is recorded as high absentee), we'd like to do a proper analysis of this criteria from all sides and we believe this could improve the model accuracy.

Question 15: Do you consider this project to be in the 'innovative category' or a 'good application' category? Provide your reason.

Yes. I consider the project to be innovative because of the following reasons

1. Transform the 721 observations to user summarised 36 observation dataset so that we could draw user-specific features, which makes the EDA relatively easy.
2. Usage of the transformed dataset to select distuingishing and self-suggestive features in a simple, hassle-free, easy to understand and effective manner
3. Attempt to use clustering to understand the feature space and deduce find the most suitable classifier(s) for the data. This would save us huge and valuable of time, particularly if we've a very big dataset which is the case mostly.
4. Succesful Demonstration of the above by pitting the selected classifier against other classifiers
5. Opens door for further new and interesting type of analysis for the same dataset as mentioned for question 14.