Happiness Classification Dataset

1) Importing required libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

2) Let's import the dataset `happydata.csv` and explote it. Check the structure of the data by inspecting the number of rows, columns, and attributes of the dataset. You can use functions like .shape, .head(), .info(), and .describe() to get a quick overview of the data.

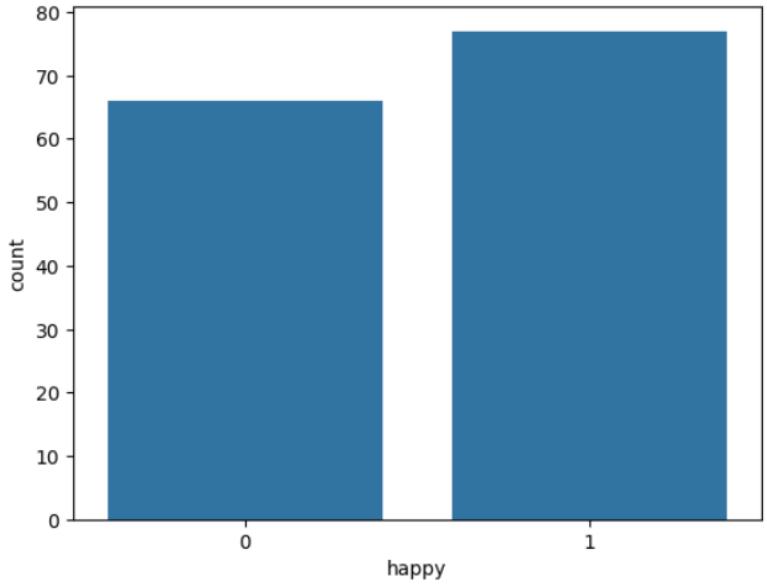
```
df = pd.read_csv('happydata.csv')
df.head()
```

	infoavail	housecost	schoolquality	policetrust	streetquality	ëvents	happy
0	3	3	3	4	2	4	0
1	3	2	3	5	4	3	0
2	5	3	3	3	3	5	1
3	5	4	3	3	3	5	0
4	5	4	3	3	3	5	0

3) Let's plot the frecuency of each class, features scatterplot and compute the correlation matrix.

```
sns.countplot(data=df, x='happy')
```

```
<Axes: xlabel='happy', ylabel='count'>
```



4) Let's calculate the correlation matrix with the numerical variables in the notebook.

```
corr = df.corr()
corr.style.background_gradient(cmap='coolwarm')
```

	infoavail	housecost	schoolquality	policetrust	streetquality	ëvents	happy
infoavail	1.000000	0.092676	0.301971	0.104378	0.399203	0.417521	0.312740
housecost	0.092676	1.000000	0.181081	0.107432	-0.002141	0.024546	0.019368
schoolquality	0.301971	0.181081	1.000000	0.298898	0.329874	0.207006	0.163639
policetrust	0.104378	0.107432	0.298898	1.000000	0.269420	0.199151	0.113356
streetquality	0.399203	-0.002141	0.329874	0.269420	1.000000	0.307402	0.206685
ëvents	0.417521	0.024546	0.207006	0.199151	0.307402	1.000000	0.220729
happy	0.312740	0.019368	0.163639	0.113356	0.206685	0.220729	1.000000

```
df.columns
```

```
Index(['infoavail', 'housecost', 'schoolquality', 'policetrust',
       'streetquality', 'ëvents', 'happy'],
      dtype='object')
```

5) Split the target and features

```
X = df[['infoavail', 'housecost', 'schoolquality', 'policetrust', 'streetquality', 'ëvents']]
y = df[['happy']]
```

6) Use `train_test_split' to split the data into training and testing sets. Split the dataset in 80% training,20% testing and random_state=0.

> ***Store the values in the variables in*** `X_train` , `X_test` , `y_train` , `y_test` , `random_state` .

from sklearn.model_selection import train_test_split

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

7) Create an instance of the `LogisticRegression` and store the model in `lr` .

```
from sklearn.linear_model import LogisticRegression
lr= LogisticRegression()
```

8) It's time to train the LogisticRegression using the training dataset.

```
lr.fit(X_train,y_train)
```

```
/usr/local/lib/python3.11/site-packages/sklearn/utils/validation.py:1310: DataConversionWarning: A column-vector y was passed w
hen a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

LogisticRegression

```
LogisticRegression()
```

9) Use the trained model to make predictions on the test data

```
y_pred = lr.predict(X_test)
```