```
import pandas as pd
```

```
iris_names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'spe
iris = pd.read_csv('https://gist.githubusercontent.com/curran/a08a1080b88344b0c8
iris.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

```
# Description of the iris dataset
iris.describe()
```

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
# Specify the targets and the features (inputs == features || outputs == targets
iris['species'] = iris['species'].replace({'setosa':0,'versicolor':1,'virginica'
iris_targets = iris['species']

iris_features = iris.drop(iris_targets)
```

```
# Mean normalization of the inputs
iris_features_normalized = (iris_features - iris_features.mean()) / iris_feature
# Show first instances
iris_features_normalized.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
3	-1.530460	0.113830	-1.319718	-1.345671	-1.25228
4	-1.045595	1.259928	-1.376911	-1.345671	-1.25228
5	-0.560729	1.947587	-1.205332	-1.081568	-1.25228
6	-1.530460	0.801489	-1.376911	-1.213619	-1.25228
7	-1.045595	0.801489	-1.319718	-1.345671	-1.25228

```
# Min max normalization
uci_wine_inputs_minmax = (iris_features - iris_features.min()/(iris_features.max
uci_wine_inputs_minmax.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
3	3.405556	2.266667	1.330508	0.158333	0.0
4	3.805556	2.766667	1.230508	0.158333	0.0
5	4.205556	3.066667	1.530508	0.358333	0.0
6	3.405556	2.566667	1.230508	0.258333	0.0
7	3.805556	2.566667	1.330508	0.158333	0.0

```
# Finne prosenter av hver instanse i klassene
iris.groupby('species').count()/len(iris_features)
# Through series
iris_targets.value_counts()/len(iris_targets)
```

```
0    0.333333
1    0.333333
2    0.333333
Name: species, dtype: float64
```

```
# Split 60 / 20 / 20
random_train = iris.sample(frac=0.6)
random_val_test = iris.drop(random_train.index)
random_val = random_val_test.sample(frac=0.5)
random_test = random_val_test.drop(random_val.index)

random_test['species'].value_counts()/len(random_test)
```

```
1    0.366667
0    0.333333
2    0.300000
Name: species, dtype: float64
```

```
# stratified
stratified_train = iris.groupby('species', group_keys=False).apply(lambda x: x.s
stratified_val_test = iris.drop(random_train.index)
stratified_val = random_val_test.groupby('species', group_keys=False).apply(lamb
stratified_test = random_val_test.drop(random_val.index)

stratified_test['species'].value_counts()/len(random_test)
```

```
1    0.366667
0    0.333333
2    0.300000
Name: species, dtype: float64
```

```

```

Colab paid products - Cancel contracts here