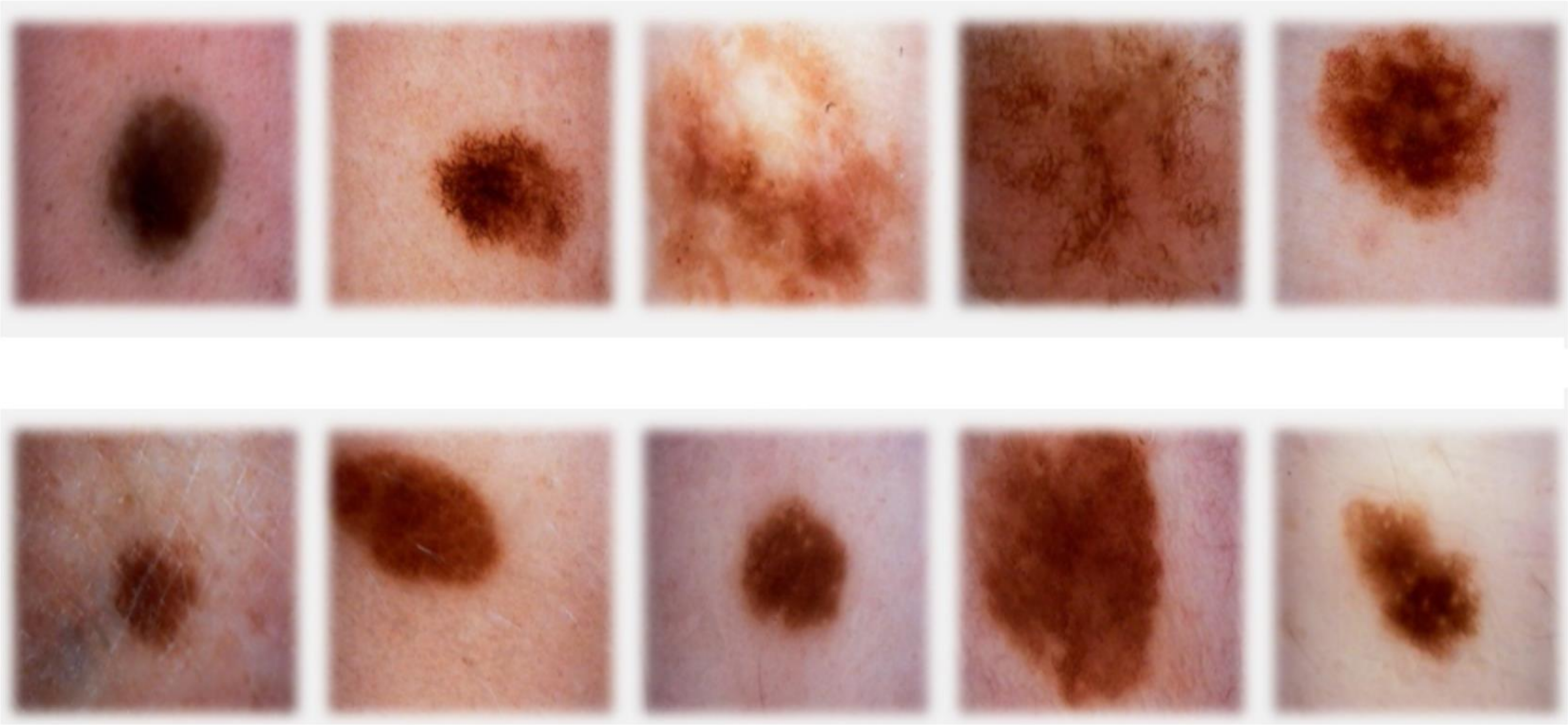
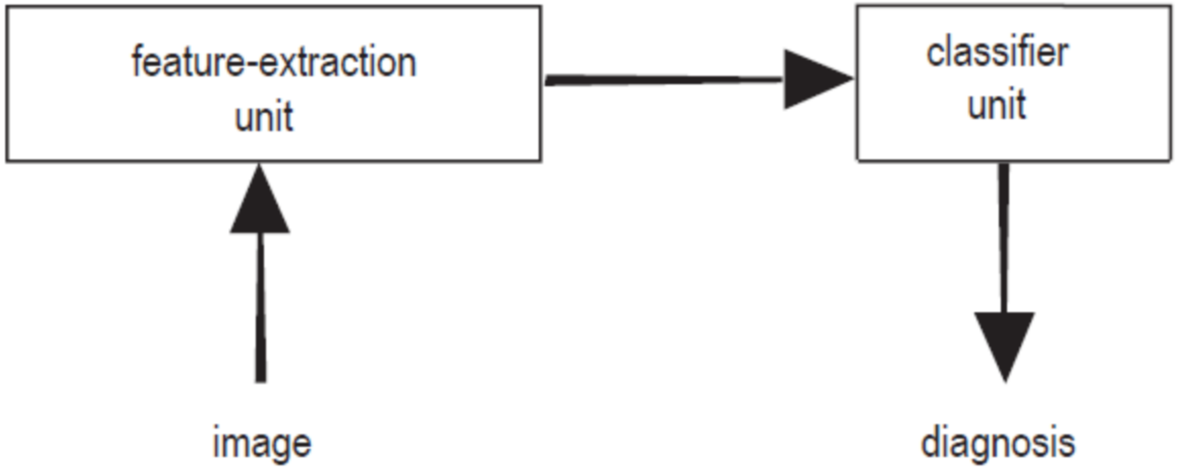
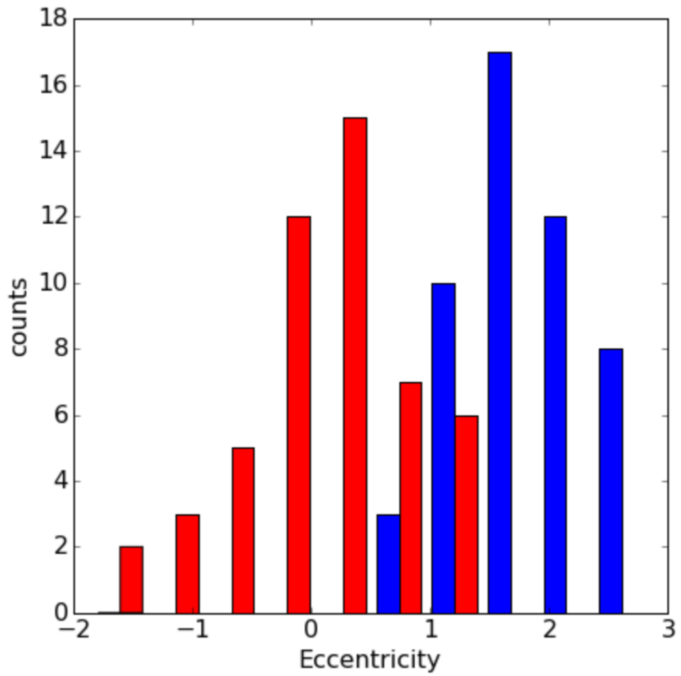
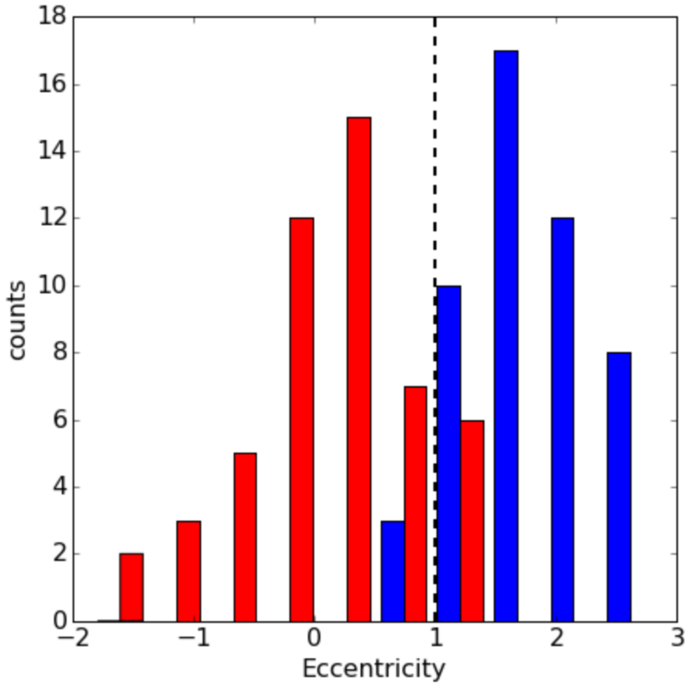


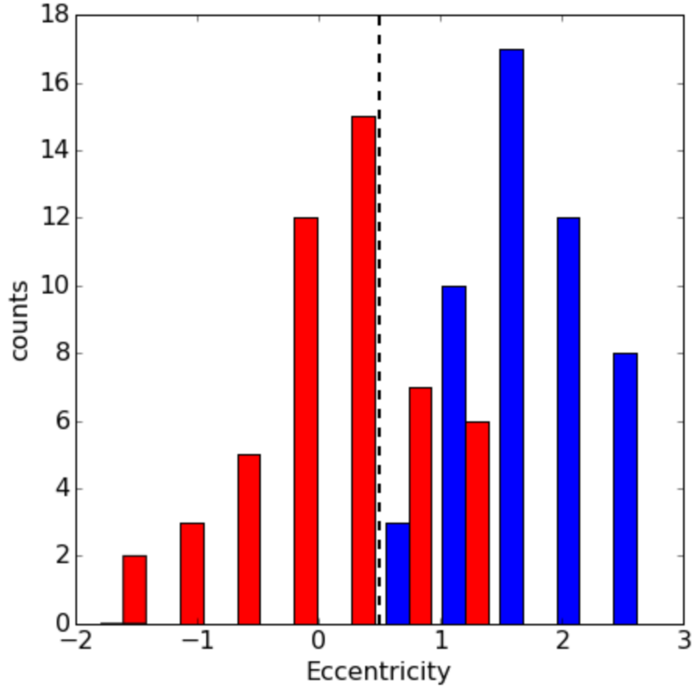
- sign of cancer
- top row malignant
- bottom row benign



- feature extraction: features (a.k.a. properties or attributes)
- data set, sample (a.k.a. example, instance or data point), label (a.k.a. target)







- eccentricity of lesion (how nearly circular the lesion is)
- threshold ***t***
- consequence of the predictions

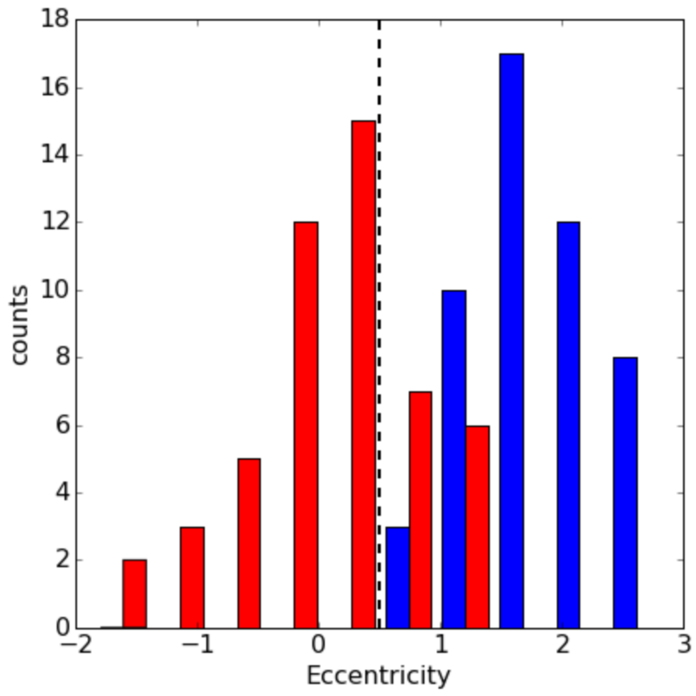
- malignant positive class, benign negative
- consequence of the predictions

count the number of malignant images with eccentricity value $\geq t$: **true positive** predictions (TP)

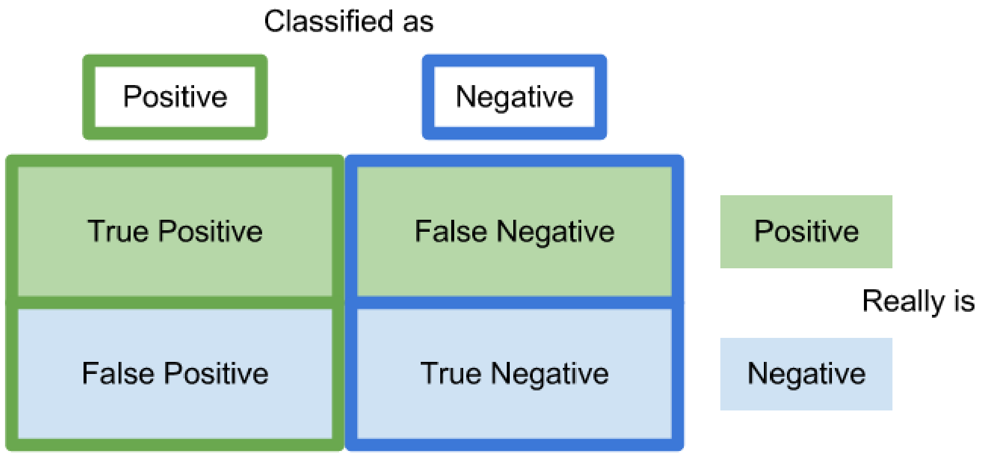
count the number of malignant images with eccentricity value $< t$: **false negative** predictions (FN)

count the number of benign images with eccentricity value $\geq t$: **false positive** predictions (FP)

count the number of benign images with eccentricity value $< t$: **true negative** predictions (TN)



	Classified as: Positive	Classified as: Negative
Really is: Positive	True Positive	False Negative
Really is: Negative	False Positive	True Negative

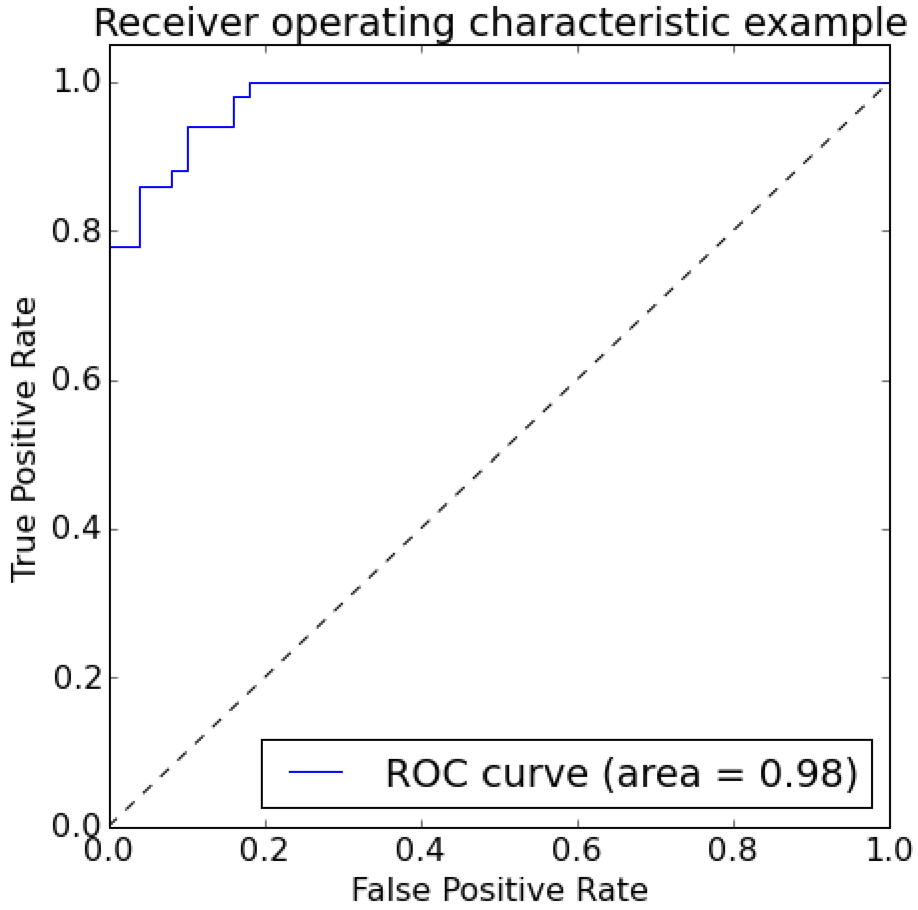


$$\text{accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

$$\text{TPR} = \frac{TP}{TP + FN}$$

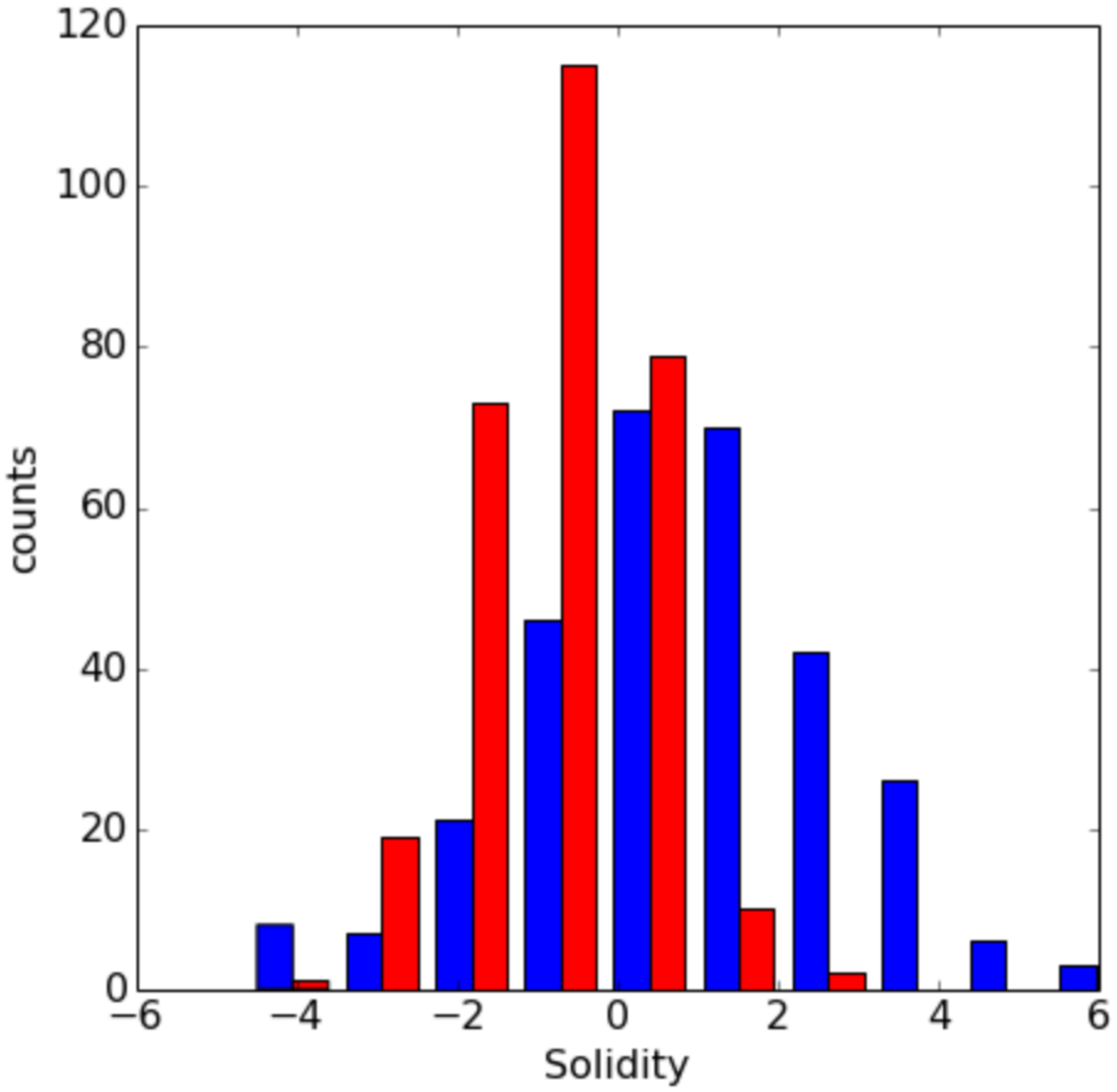
$$\text{FPR} = \frac{FP}{FP + TN}$$

$$\mathrm{TPR} = \frac{TP}{TP + FN}$$

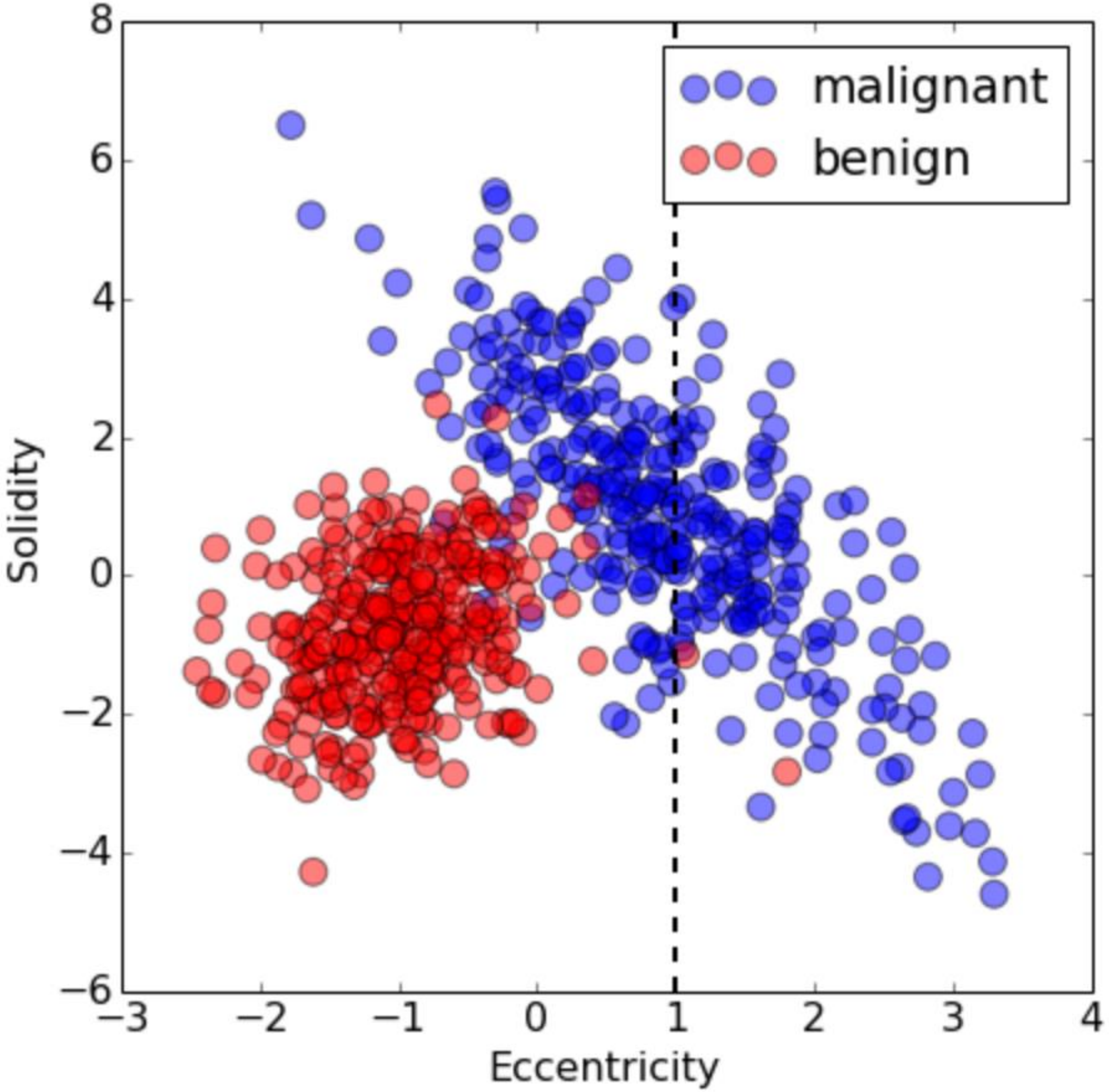


$$\mathrm{FPR} = \frac{FP}{FP + TN}$$

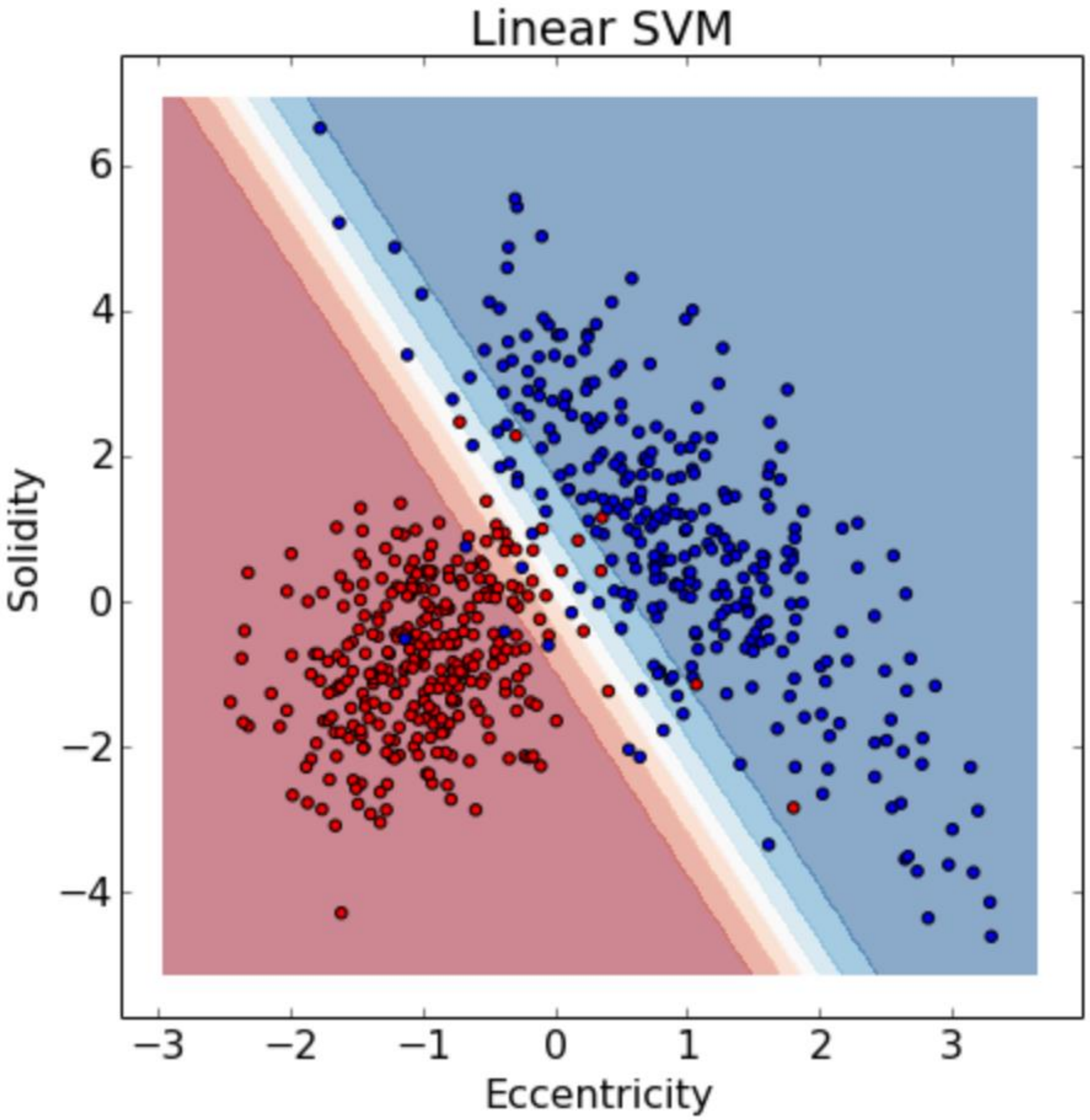
- model that classifies all images as malignant: TPR=1 and FPR=1
- model that classifies all images a benign: TPR=0 and FPR=0
- vary threshold *t*
- *AUC*



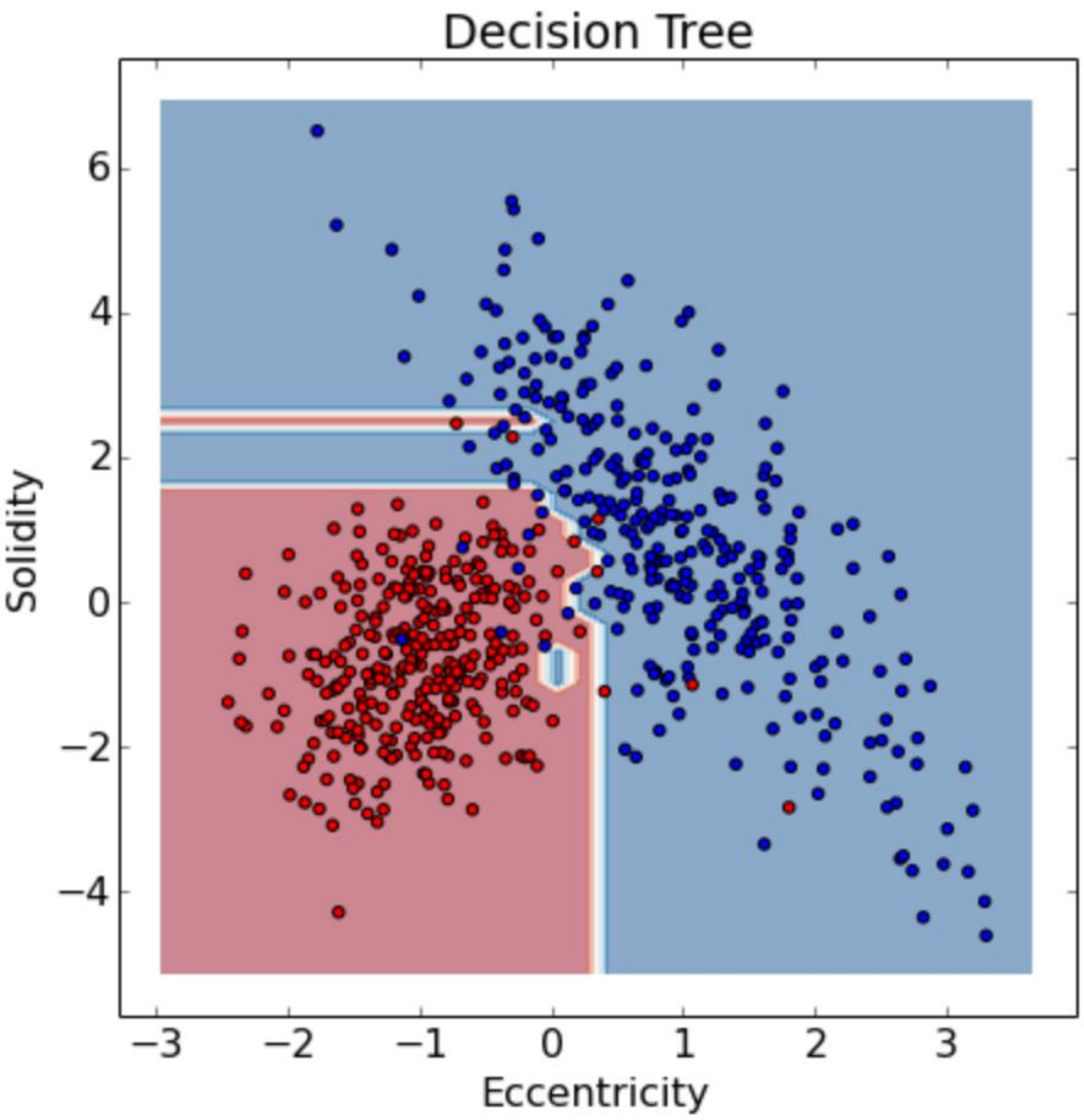
- add another feature?
- feature vector X
- Euclidean vector space



- feature vector X
- Euclidean vector space



- linear decision boundary
- blue region malignant, red region benign
- yet more features
- can’t look at the decision boundary
- more complex



- unseen external images
- generalization
- overfitting

data normalization

- make all features same scale
- Eccentricity [0,100], Solidity [-5,7]
- weights all features equally in their representation
- **standardization**

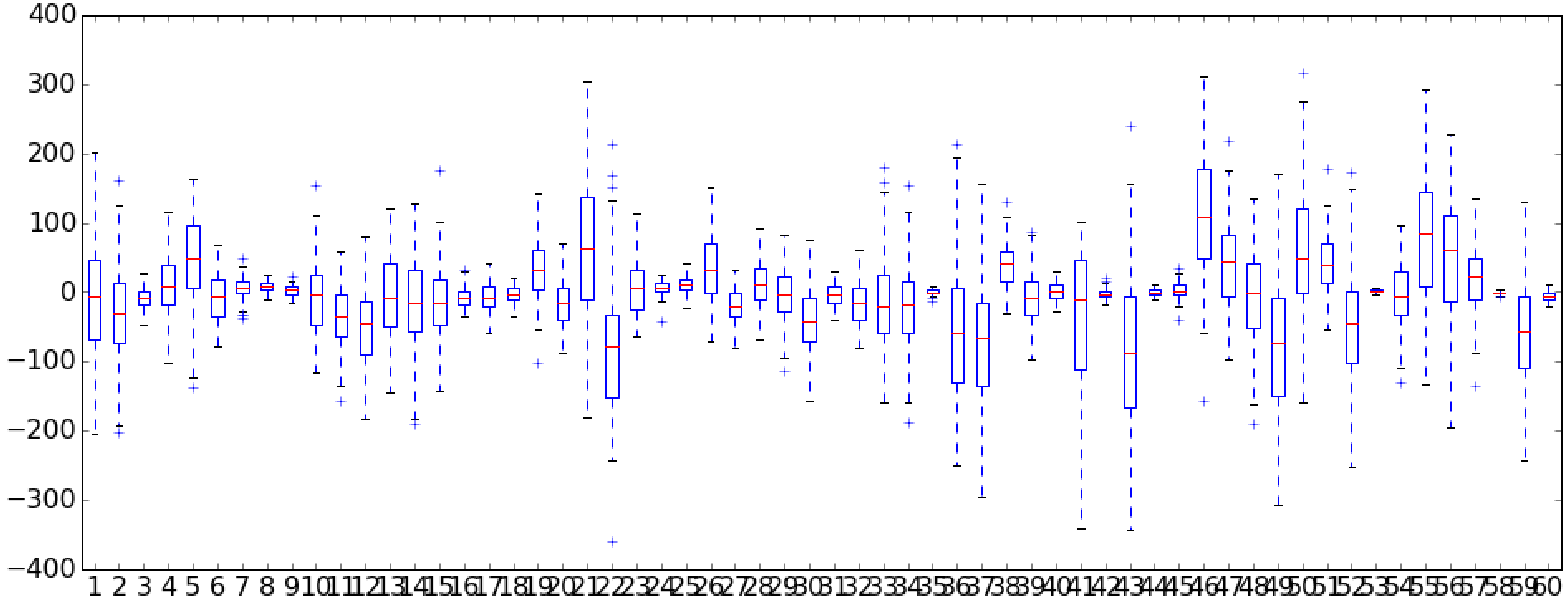
$$\mu = 0 \quad \sigma = 1$$

- **min-max scaling**: scale the features to a fixed range

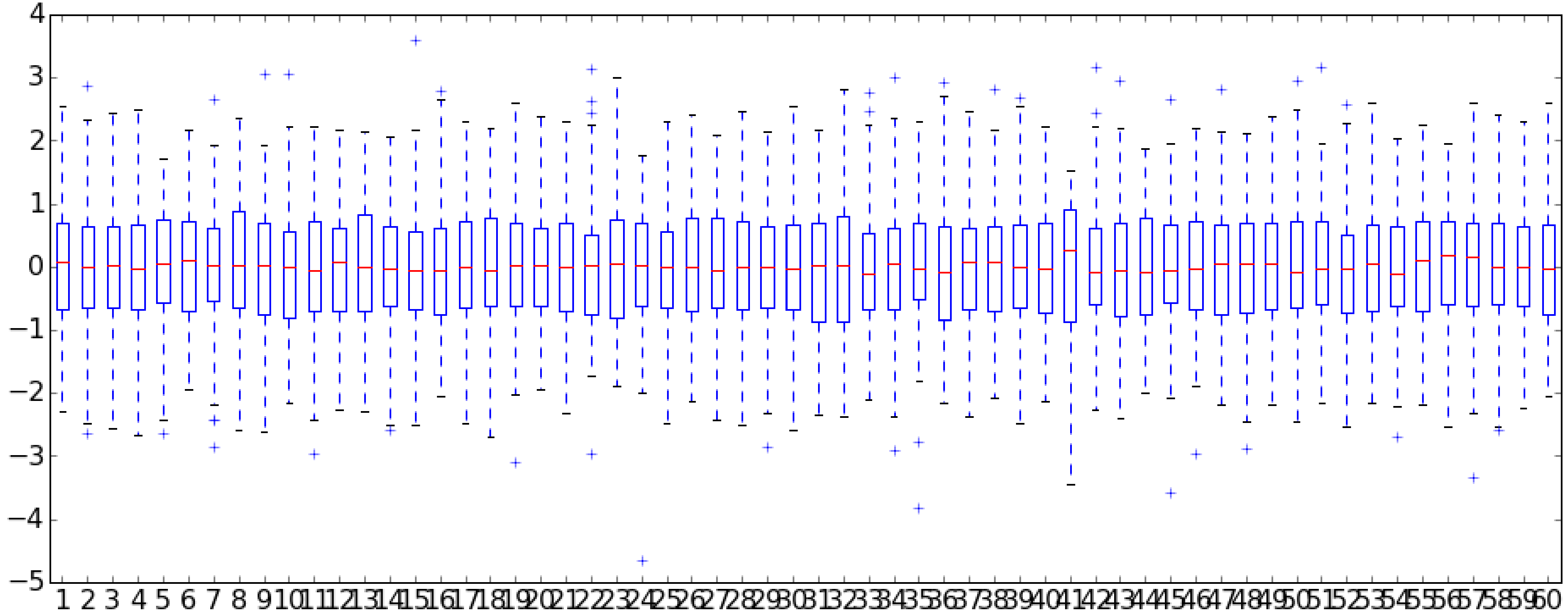
$$x_{norm} = \frac{x - \mu}{\sigma}$$

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

data normalization

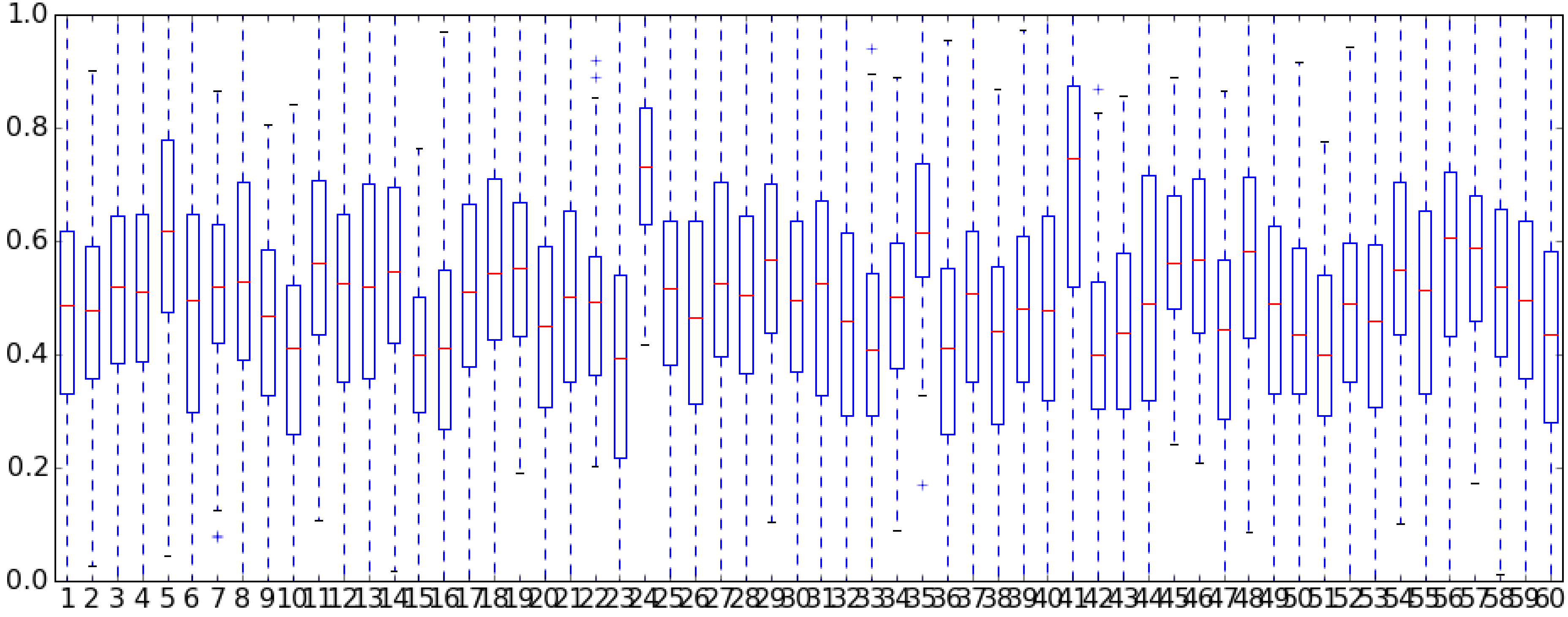


data normalization



$$x_{norm} = \frac{x - \mu}{\sigma}$$

data normalization



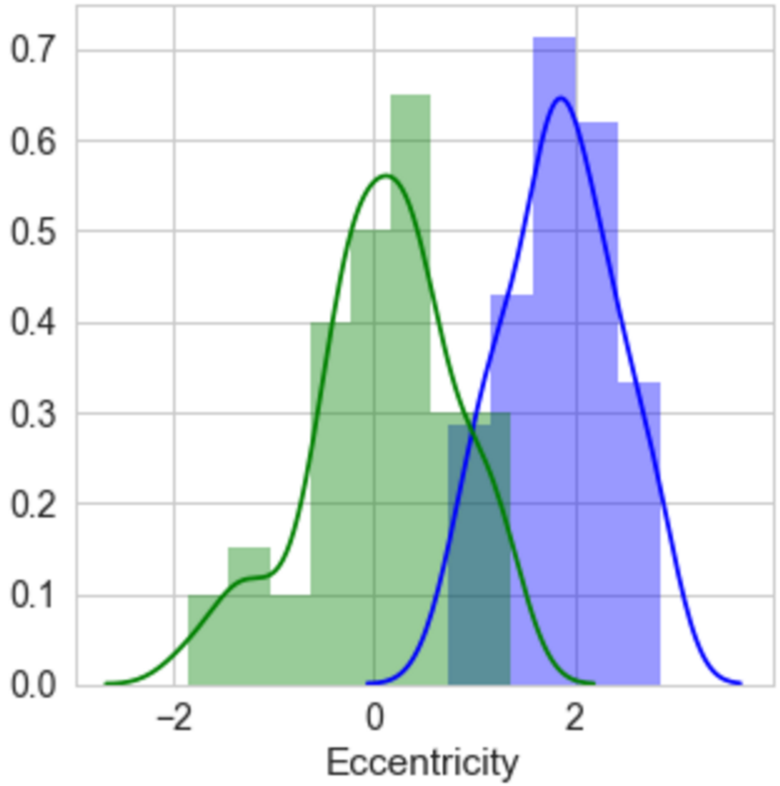
$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

```
%matplotlib inline
import matplotlib.pyplot as plt;
import seaborn as sns;
sns.set_context("notebook", font_scale=1.4);
sns.set_style("whitegrid");

import numpy as np;
import pandas as pd;

import imp;
compomics_import = imp.load_source('compomics_import', '../compomics_import.py');

from IPython.core.display import HTML;
css_file = '../my.css';
HTML(open(css_file, "r").read())
```

```
dataset_ecc = pd.read_csv("eccentricity.csv",sep='\t')
print "First 5 data points:"
print dataset_ecc.head(5)
print dataset_ecc.shape
```

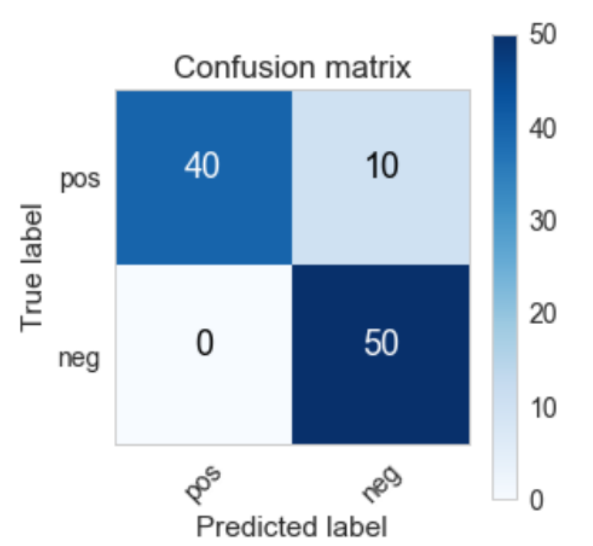
```
First 5 data points:
   Eccentricity  label
0      1.770190      1
1      0.952262      0
2      1.726489      1
3      0.344707      0
4     -0.138108      0
(100, 2)
```

```
plt.figure(figsize=(5,5))
#call  seaborn to plot a distribution plot in blue
sns.distplot(dataset_ecc[dataset_ecc.label==1]['Eccentricity'], color="b")
#call  seaborn to plot a distribution plot in green
sns.distplot(dataset_ecc[dataset_ecc.label==0]['Eccentricity'], color="g")
plt.show()
```

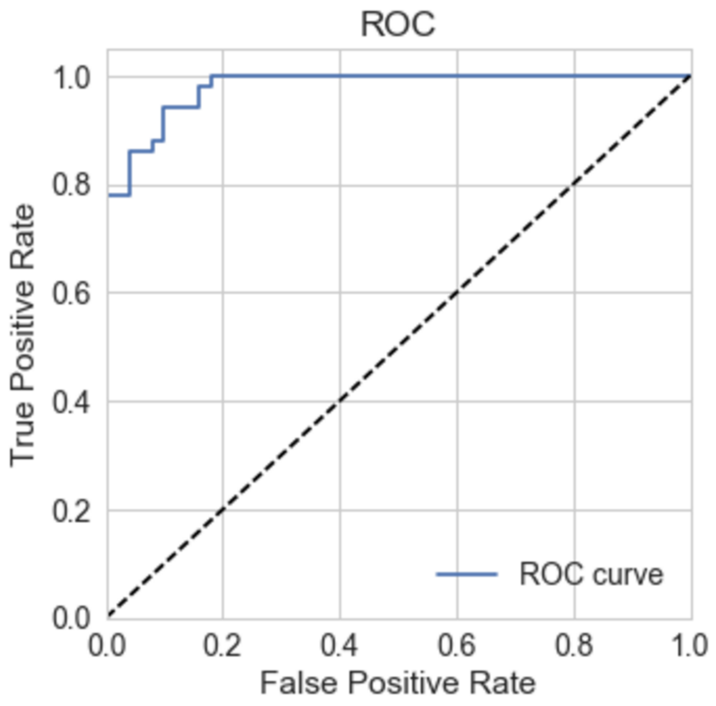


```python
from sklearn import metrics

t=0.7
predictions = [1 if x >= t else 0 for x in dataset_ecc['Eccentricity']]

cm = metrics.confusion_matrix(dataset_ecc['label'],predictions)
print "Confusion matrix computed by scikit-learn:\n"
print cm
print
print "#TP = %i" % cm[0][0]
print "#FP = %i" % cm[1][0]
print "#FN = %i" % cm[0][1]
print "#TN = %i" % cm[1][1]
```

```
Confusion matrix computed by scikit-learn:

[[40 10]
 [ 0 50]]

#TP = 40
#FP = 0
#FN = 10
#TN = 50
```



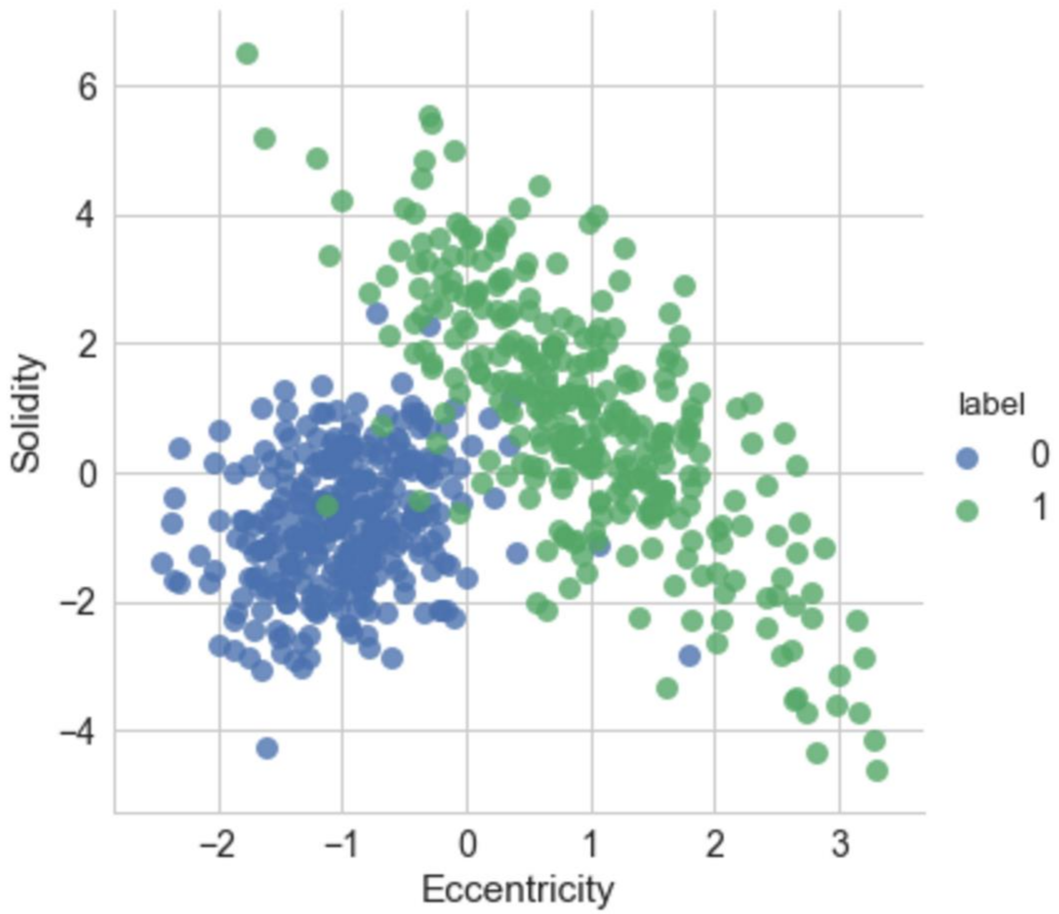
```
fpr, tpr, thresholds = metrics.roc_curve(dataset_ecc['label'],
                                          dataset_ecc['Eccentricity'], pos_label=1)

plt.figure(figsize=(5,5))
compomics_import.plot_ROC(fpr,tpr)
plt.show()
```



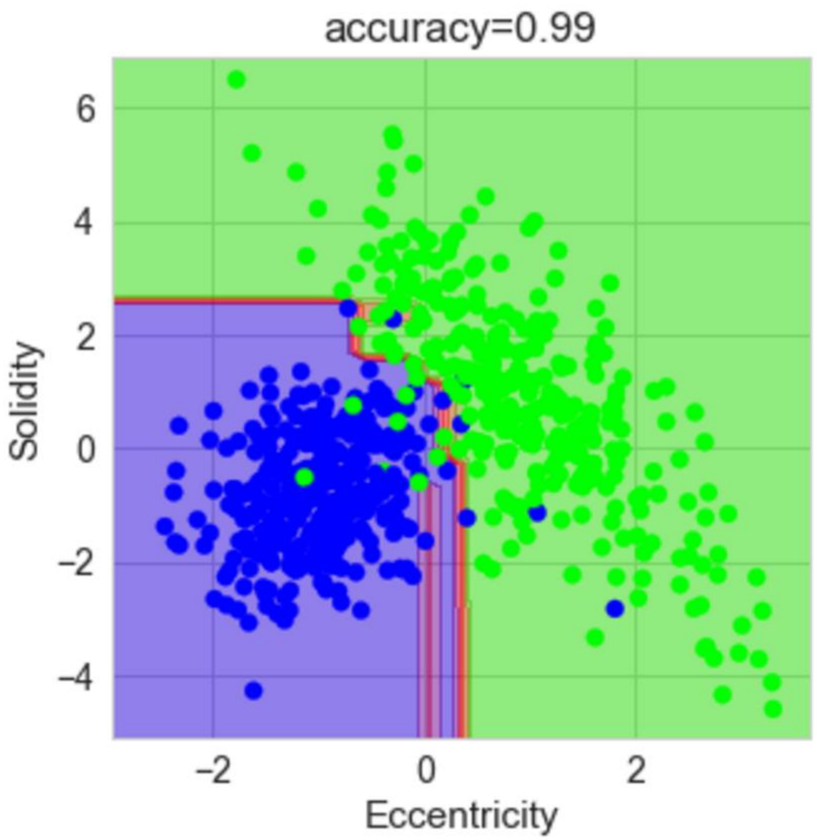

```
print metrics.roc_auc_score(dataset_ecc['label'], dataset_ecc['Eccentricity'])
```

```
0.9792
```

```python
sns.lmplot(x="Eccentricity", y="Solidity", data=dataset_mel, hue='label',
           fit_reg=False, size=5.5, scatter_kws={"s": 80})
plt.show()
```



```python
from sklearn.tree import DecisionTreeClassifier

model =  DecisionTreeClassifier(max_depth=5,random_state=0)
model.fit(X, y)
plt.figure(figsize=(5,5))
compomics_import.plot_decision_boundary(model,X,y)
plt.show()
```



```
dataset_unscaled = pd.read_csv('unscaled_dataset.csv',sep=',')
plt.figure(figsize=(12,6))
dataset_unscaled.boxplot(vert=False)
plt.show()
```