Lab 3. Logistic Regression

Intro to Machine Learning
Fall 2018, Innopolis University

Lecture recap

- Extension of linear regressions
  - Interaction
  - Polynomial
- Classification
- Logistic Regression
- Confusion Metric

Questions about the lecture

Was the material already familiar to you?

What new things have you learned?

What was hard to understand?

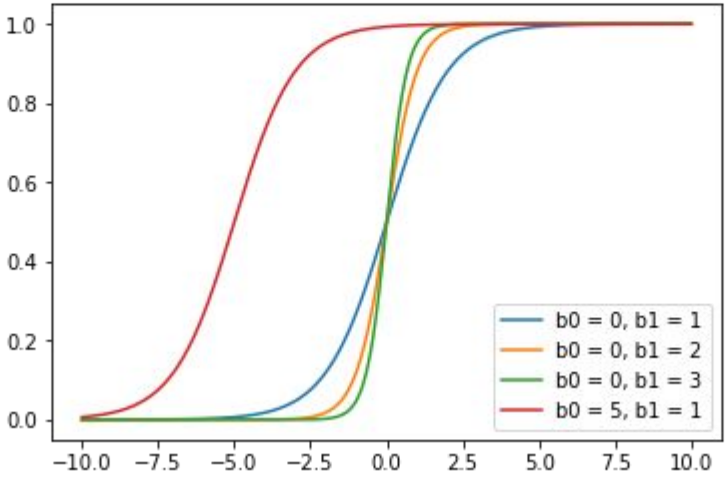
Binary classification

$$y \in \{0, 1\}$$

We will estimate the probability for the class 1

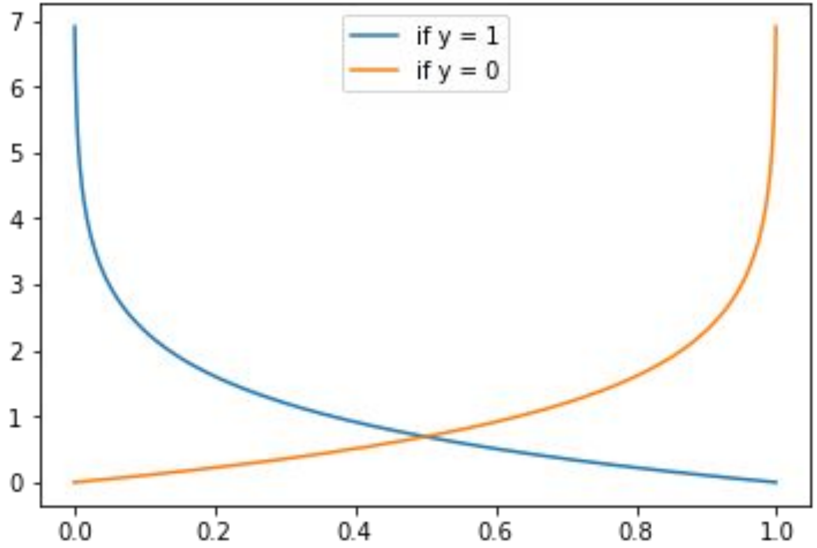
Logistic Regression

$$\hat{p}(x) = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$$



$$\hat{y} = \begin{cases} 1 & \hat{p}(x) > threshold \\ 0 & othwrwise \end{cases}$$

Cost Function



$$L(\hat{p}(x_i), y_i) = \begin{cases} -\log(\hat{p}(x_i)), & y_i = 1 \\ -\log(1 - \hat{p}(x_i)), & y_i = 0 \end{cases}$$

$$L(\hat{p}(x_i), y_i) = -y_i \log(\hat{p}(x_i)) - (1 - y)_i \log(1 - \hat{p}(x_i))$$

Find derivations

Given

$$\sigma(z) = \frac{e^z}{1+e^z}$$

$$\hat{p}(x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}$$

$$\frac{\partial\sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z))$$

$$L(\hat{p}(x_i), y_i) = -y_i \log(\hat{p}(x_i)) - (1 - y)_i \log(1 - \hat{p}(x_i))$$

Find

$$\frac{\partial L}{\partial \beta_0}, \frac{\partial L}{\partial \beta_1}$$

Gradient Descent

$$\frac{\partial L}{\partial \beta_0} = \sigma(z) - y$$
$$\frac{\partial L}{\partial \beta_1} = x(\sigma(z) - y)$$

How to use partial derivatives in a gradient descent?

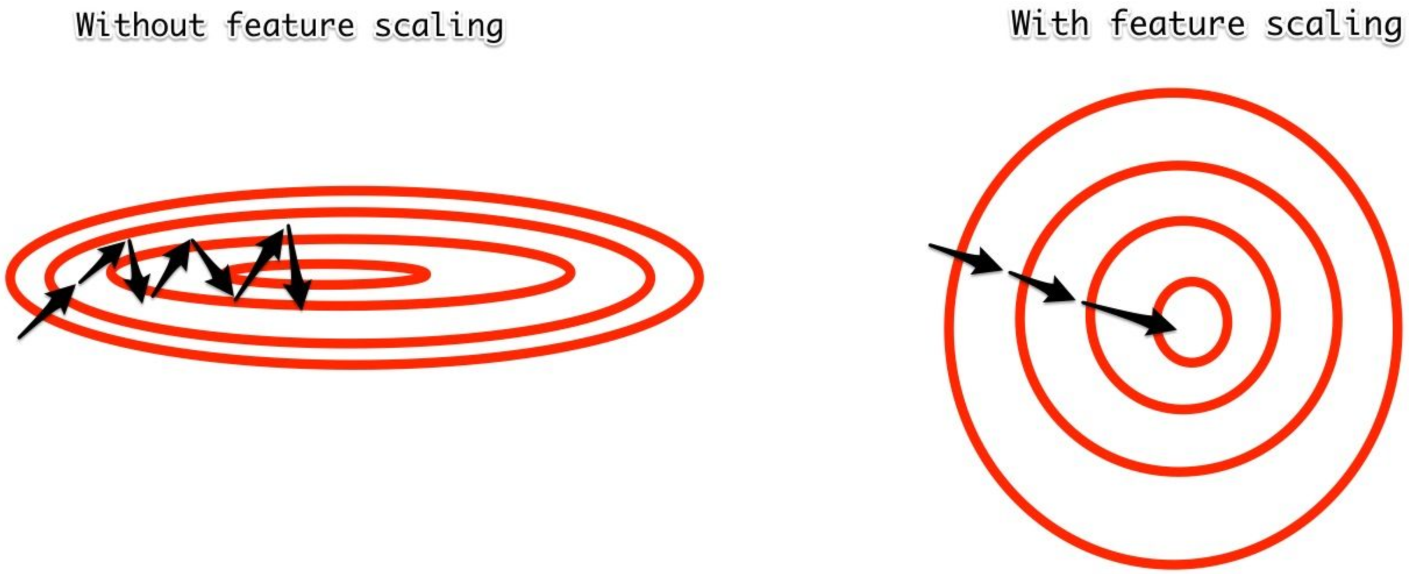
Gradient Descent

$$
\begin{aligned}
\frac{\partial L}{\partial \beta_0} &= \sigma(z) - y \\
\frac{\partial L}{\partial \beta_1} &= x(\sigma(z) - y)
\end{aligned}
$$

How to use partial derivatives in a gradient descent?

$$
\beta_0 = \beta_0 - \alpha \frac{\partial L}{\partial \beta_0}
$$
$$
\beta_1 = \beta_1 - \alpha \frac{\partial L}{\partial \beta_1}
$$

Feature Scaling and GD



https://www.kaggle.com/jannesklaas/ai-bootcamp-9-feature-data-prep

Feature Scaling

$-1 \leq x_1 \leq 1$	OK
$0 \leq x_2 \leq 1$	OK
$0 \leq x_3 \leq 3$	OK
$-1000 \leq x_4 \leq 1000$	X
$-0.001 \leq x_5 \leq 0.001$	X

Feature Scaling

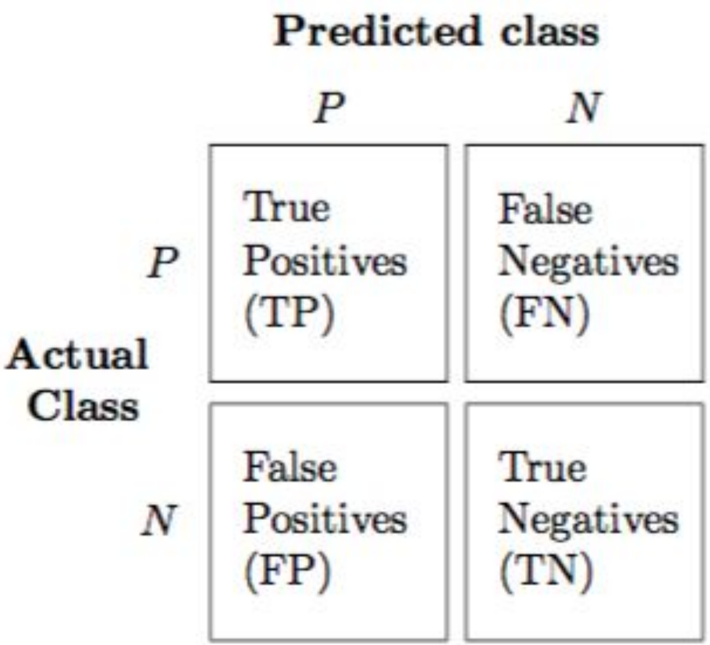
Get every feature into approximately a -1 <= x <= 1 range

$$x' = \frac{x - mean}{max - min}$$

Or

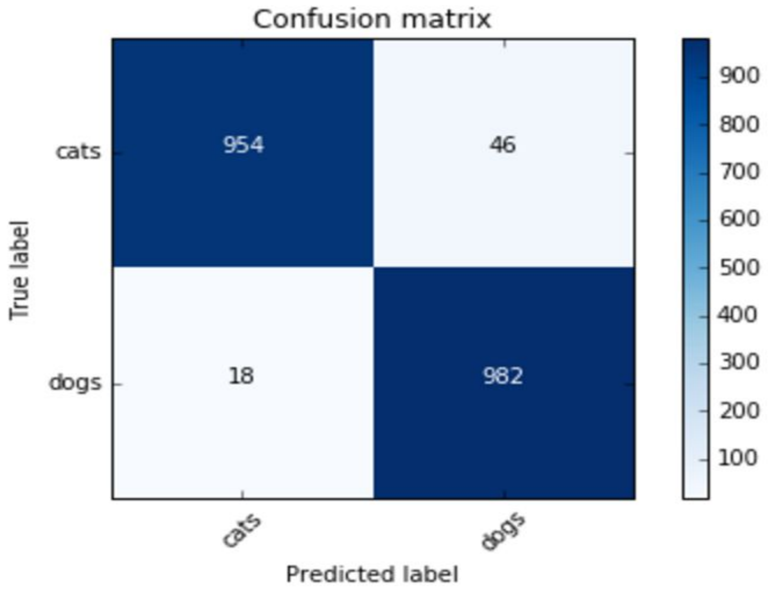
$$x' = \frac{x - \mu}{\sigma}$$

Confusion Matrix



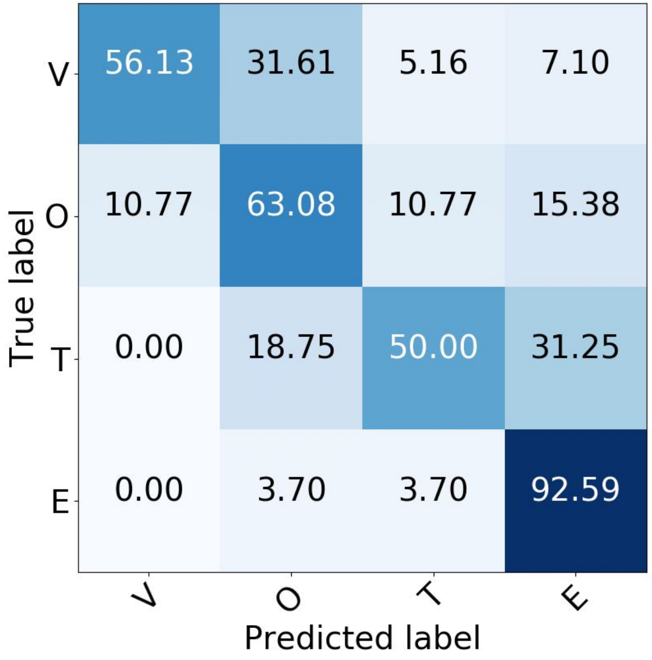
https://rasbt.github.io/mlxtend/user_guide/evaluate/confusion_matrix/

Confusion Matrix



http://wiki.fast.ai/index.php/Lesson_2_Notes

Confusion Matrix



https://www.researchgate.net/figure/Normalized-confusion-matrix-of-best-performing-models-on-devel-subset-a-SVM_fig2_324226324

Recall and Precision

$$\hat{y} = \begin{cases} 1 & \hat{p}(x) > threshold \\ 0 & othwrwise \end{cases}$$

$$Precision = \frac{True\ positives}{True\ positives + False\ positives}$$

$$Recall = \frac{True\ positives}{True\ positives + False\ negatives}$$

Classification Threshold

Changing the threshold we are getting

different recall and precision.

How to show the model quality

without a free parameter?

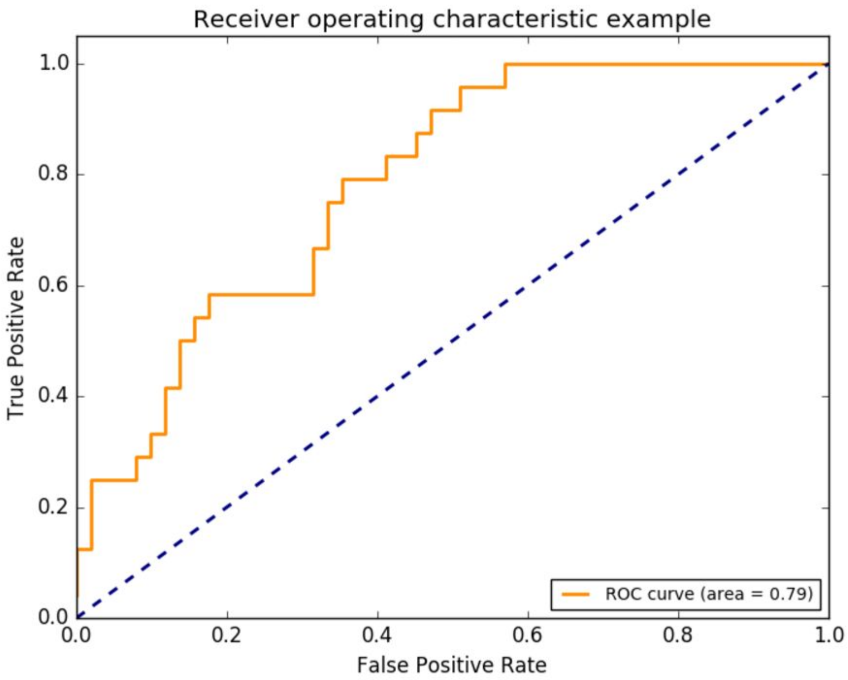
ROC Curve

Changing the threshold we are getting

different recall and precision.

How to show the model quality

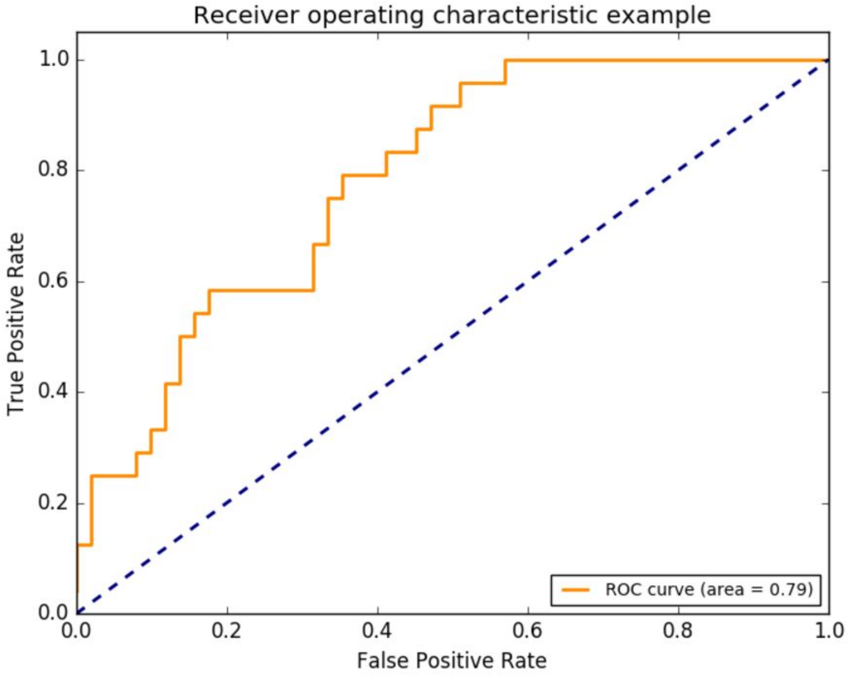
without a free parameter?



https://qiita.com/bmj0114/items/460424c110a8ce22d945

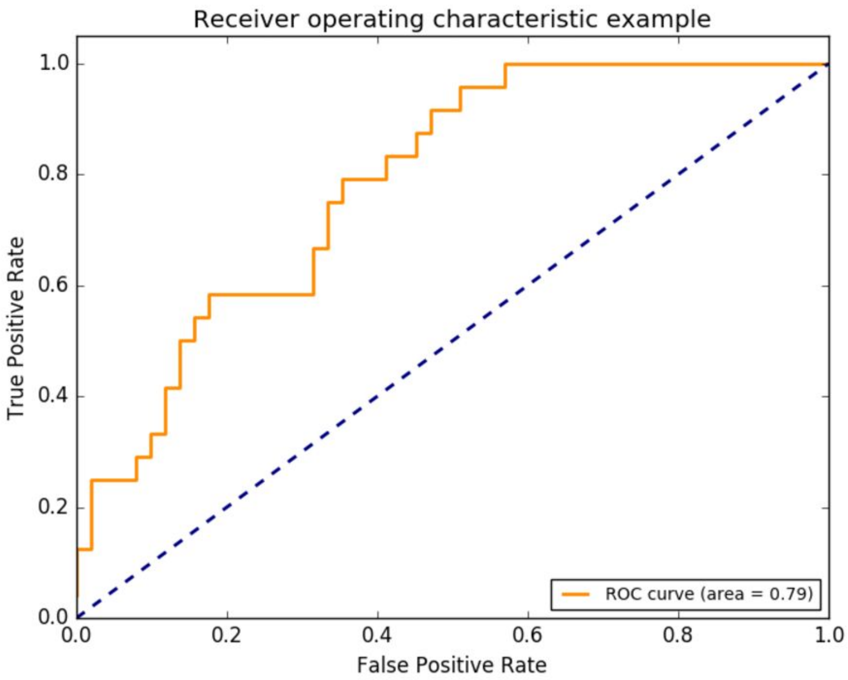
ROC Curve

What are the properties of the curve?

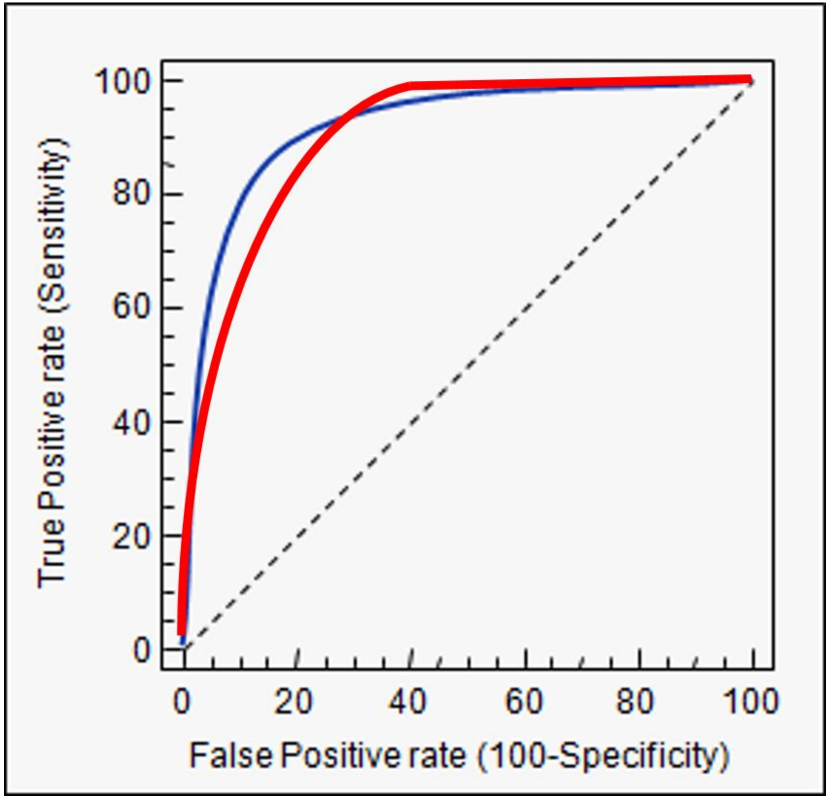


ROC Curve

How to select a threshold?



ROC Curve



Which model is better?

https://stats.stackexchange.com/questions/264477/will-roc-curve-for-a-model-always-be-symmetric-if-we-have-enough-training-data

Exercise

1. Download iris dataset (from the first lab or import form sklearn datasets)
2. Select one feature and two out of three classes
3. Split data to the train and test parts
4. Try your GD for logistic regression on this data
5. Measure accuracy for selected threshold

Sklearn Logistic Regression

```
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

iris = load_iris()
X = iris.data
y = iris.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_pred, y_test)
print(accuracy)
```

HW Task 1

Finalize GD for a logistic regression

HW Task 2

1. Download the Kickstarter projects dataset
   https://www.kaggle.com/kemical/kickstarter-projects
2. Select a columns for a logistic regression. Do the necessary preprocessing
3. Split data to the test and train parts
4. Remove all cancelled projects
5. Predict the probability of the success by the sklearn Logistic regression
6. Calculate the recall and precision for your model
7. Train the logistic regression for all data, including cancelled projects as well
8. Calculate the recall and precision for the new model

That’s it for today! Questions?