知能情報実験3：データマイニング班
(week 3) 線形回帰モデルと最急降下法

1. 復習
2. scikit-learn入門
3. モデルとは？（問題設定、アルゴリズム、モデル）
4. 線形回帰モデル
5. 仮説、損失関数、目的関数
6. 最小二乗法
7. 最急降下法
8. 参考サイト

実験ページ: http://ie.u-ryukyu.ac.jp/~tnal/2020/info3/dm/

Example: *Iris* flower data set

review

http://en.wikipedia.org/wiki/Iris_flower_data_set

(1) What is experience E?
(2) What is task T?
(3) How to measure the performance P?

- **Classification**
  - In Classification, the samples belong to two or more classes and we want to learn from already labeled data how to predict the class of unlabeled data.
  - E.g., distinguishes the species from each other.
  - Dataset = samples vs. features and classes

- Input data, X
- 4 features or attributes

- Teach data
- supervisory signal
- output data, Y
- target
- 1 class in 3 classes

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	*I. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*
5.0	3.6	1.4	0.2	*I. setosa*

1 sample

Example: boston house prices dataset

http://archive.ics.uci.edu/ml/datasets/Housin

review

(1) What is experience E?
(2) What is task T?
(3) How to measure the performance P?

- Regression
  - If the desired output consists of one or more continuous variables, then the task is called *regression*.
  - E.g., concerns housing values in suburbs of Boston.
  - Dataset = samples vs. features and continuous variables

13 features

Continuous variable

CRIM	ZN	INDUS	（中略）	LSTAT	MEDV
6.32E−03	1.80E+01	2.31E+00		4.98E+00	24.00
2.73E−02	0.00E+00	7.07E+00		9.14E+00	21.60
2.73E−02	0.00E+00	7.07E+00		4.03E+00	34.70

1 sample

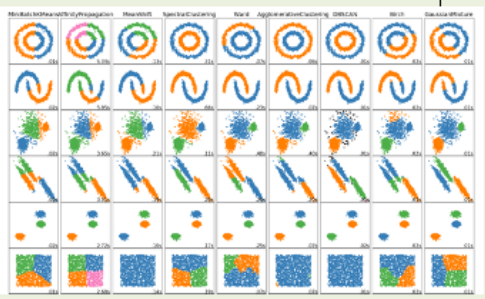
Example: *Overview of clustering methods*

https://scikit-learn.org/stable/modules/clustering.html

review

(1) What is experience E?
(2) What is task T?
(3) How to measure the performance P?

- Clustering
  - Clustering is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups (clusters).
  - Training data consists of a set of input vectors x **without any corresponding target values**.
  - Dataset = samples vs. features



Terminology

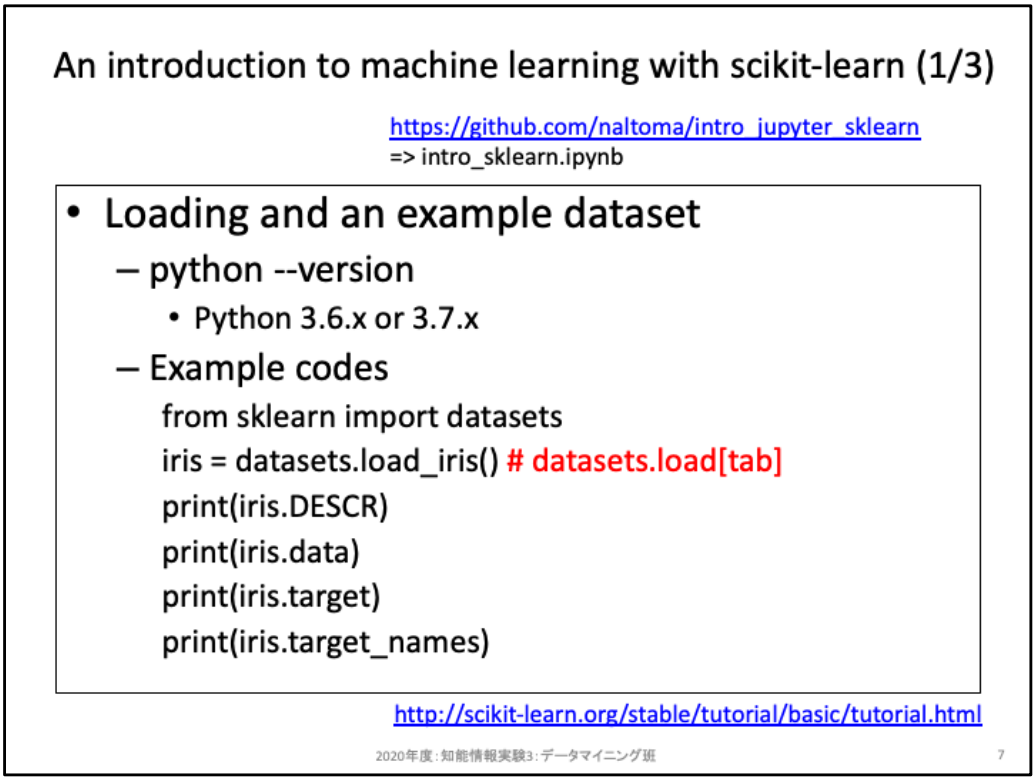
review

- ML types
  - supervised, unsupervised, semi-supervised
  - (reinforcement learning, genetic algorithm,,,)
- Task types
  - classification, regression, clustering
- sample
- features, attributes
  - numerical value
  - categorical value
  - true or false
- supervisory signal, teacher, class, label, target variable

- input, output
- Input types
  - training data / training set
  - test (for evaluation)
  - validation (for hyper params)
- model
- parameters
  - hyper parameters
  - weights, parameters
- learn, fit
- predict, estimate
- evaluation
  - open or close test
  - cross validation

知能情報実験3：データマイニング班
(week 3) 線形回帰モデルと最急降下法

1. 復習
2. scikit-learn入門
3. モデルとは？（問題設定、アルゴリズム、モデル）
4. 線形回帰モデル
5. 仮説、損失関数、目的関数
6. 最小二乗法
7. 最急降下法
8. 参考サイト

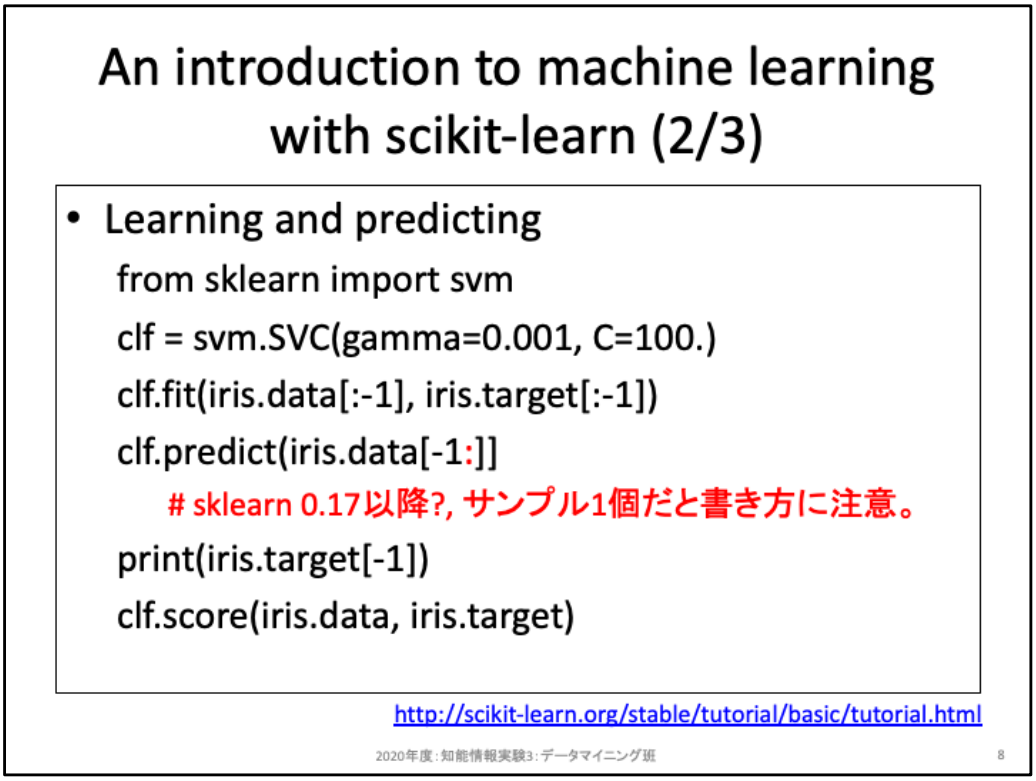


Scikit-learnにおいて機械学習用いる流れを確認してみよう。

from行はscikit-learnで用意してあるデータセットを利用するためのモジュールである。独自に用意する際には不要だが、どのように用意したら良いのかを確認するためにもここでは標準データセットを利用して進めよう。

datasets.loard_iris() で iris flower dataset を読み込むことができる。このオブジェクトはscikit-learn独自のオブジェクトであり、多くの場合は「.DESCR」でそのデータセットの詳細（str型）を確認できる。具体的なデータは「.data」と「.target」として用意されており、.dataは特徴ベクトル群、.targetは教師データ群である。これらの数（サンプル数）は同数である必要がある。「.target_names」は、教師データに関する補足である。Iris flower datasetは花の種別に関する分類タスクであり、教師データは「文字列（str型）」で用意されている。このままでは扱いにくいため、カテゴリを離散値に置き換えて利用する。どの離散値がどのカテゴリなのかが.target_namesに記載されている。

iris.dataにサンプル数分の特徴ベクトル、iris.targetにサンプル数分の教師データが用意されている。どのような型で用意されているのかはtype()関数で確認してみよう。numpy.ndarray型になっているはずだ。ndarrayのshapeも確認し、特徴ベクトルと教師データの用意の仕方を把握しよう。独自データを利用する場合にも、ndarray型で行列（のようにみえる）データセットとして特徴ベクトルを用意し、それに対応した教師データを同数用意することになる。



全体の流れとしては、(1)モデルを用意し、(2)モデルに対しデータセットを与えて学習し、(3)テスト用データセットで検証、となる。「モデルを用意する」は、ここでは何かしらブラックボックスを用意するものと捉えよう。

From行は、機械学習のモデルsvmモジュール（support vector machines）を読み込んでいる。詳細は書きを参照しよう。Scikit-learnに限らず、各モジュールの詳細は公式ドキュメントを参照するようにしよう。
https://scikit-learn.org/stable/modules/classes.html#module-sklearn.svm

2行目の svm.SVC は、SVMにおける「C-Support Vector Classification」と呼ばれるモデル（学習器）であり、「C」はドキュメントによると「Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.」とある。詳細は後日やるとして、ここでは何かを調整するパラメータのことだと理解しておこう。SVCもそうだが、一般的にモデルは「データセットを元に自動で調整する（＝学習する）パラメータ」と、「モデルを利用するユーザが手動なりで調整する必要がある（＝学習できない）パラメータ」の2種類のパラメータがある。ここで gamma, C を引数として指定しているが、このようにモデルを用意する際に指定するパラメータは手動パラメータであり、ハイパーパラメータと呼ばれる。2行目の時点ではモデルを用意しただけだ。

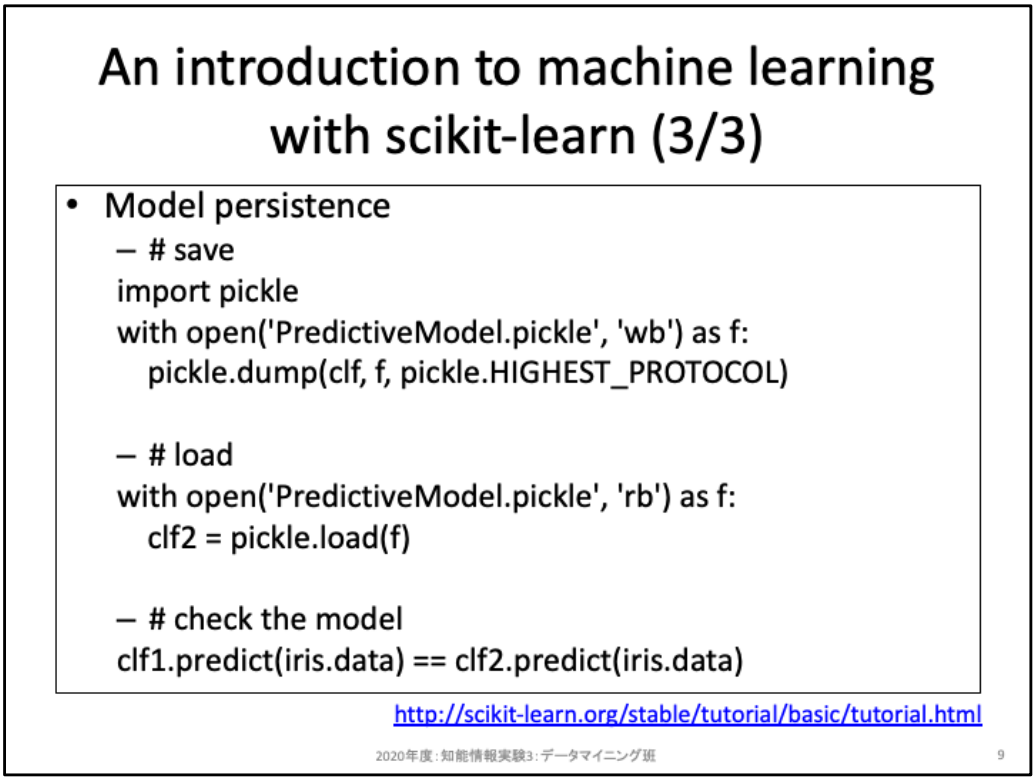
3行目の clf.fit() は、用意したモデルをデータセットに適用することで、学習させている。データセットの[:-1]までと指定しているのは、「最後の1個を残して学習用に使う」ためである。実際には「半分を学習用に、残りをテスト用に」等、適宜変更して構わない。重要な点は、教師あり学習ならば特徴ベクトル集合と教師データ集合を分けて与えるという点である。

4行目の clf.predict() は、学習したモデルを用いて予測結果を確認しようとしている。ここでは予測したいだけなので、教師データは与えていないし、与えてはいけない。

5行目の print は、実際のデータセットにおける教師データを出力しており、4行目の予測結果が適切なのか確認するために出力させている。

6行目の clf.score() は、引数に特徴ベクトル集合と教師データ集合を与えることで「スコア」を算出している。ここでスコアとは平均精度のことだが、モデルにより求め方が異なるため、ドキュメントを確認するようにしよう。

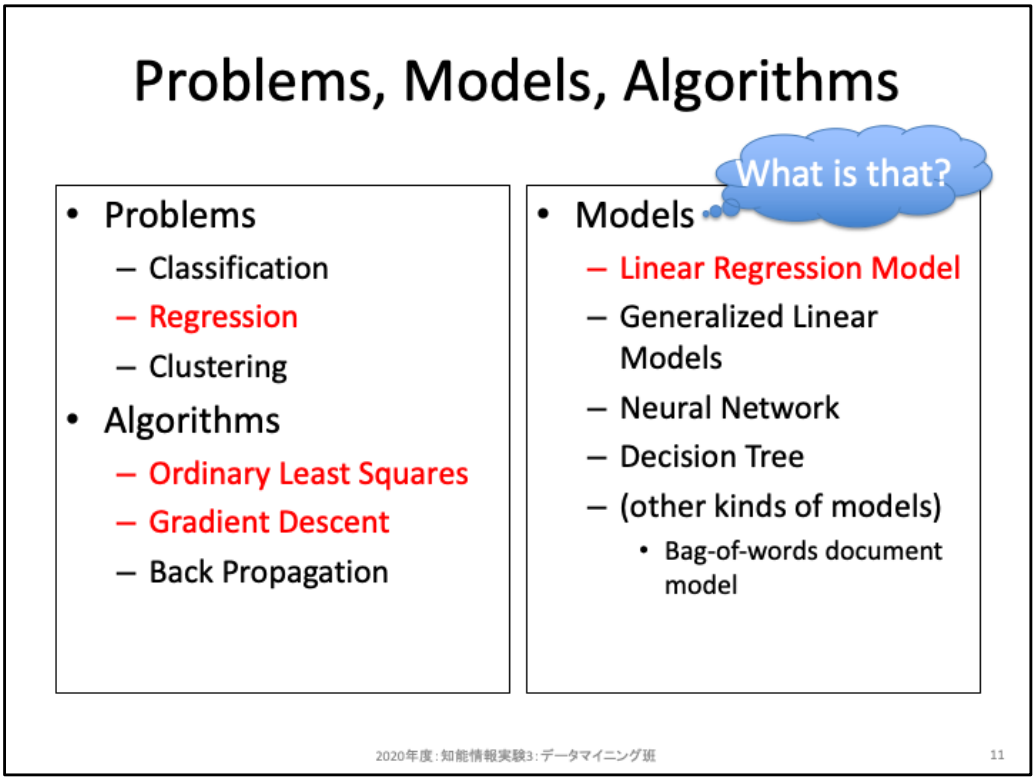
流れを振り返ると、(1)データセットを用意し、(2)モデルを用意したら、(3)model.fit関数により学習させ、(4)model.predictにより予測したりmodel.scoreによってスコアを確認することができる。(3)以降は全てのモデルに共通しているため、どのようにデータセットを用意するか、度のモデルを選択するかだけは悩む必要があるが、それ以降のコードは共通して使うことができる。



データセットやモデルが小規模ならば学習時間（リソース使用量）は気にならないが、規模が大きくなるにつれ「過去の学習結果を何度も使いまわしたい」という状況に近づいていく。そもそもアプリ等として利用する際にはそこで学習するのではなく、学習済みモデル（model.fitし終えたモデル）を用いて予測するだけという状況のほうが多い。このように学習済みモデルを保存するためには、そのパッケージで用意されている独自設計の保存モジュールを利用するか、もしくはPython標準のPickleモジュールを利用することになる。上記コードはPickleを用いてモデルをファイルに保存し、その後ファイルからモデルを読み込む例である。学習した後のモデルを用いた model.score, model.predict と、保存済みモデルから復元したモデルを用いた score, predict とが同じであること確認してみよう。

情報工学実験4：データマイニング班
(week 3) 線形回帰モデルと最急降下法

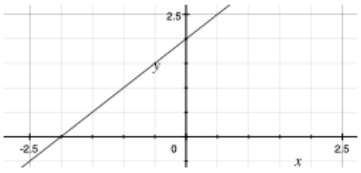
1. 復習
2. scikit-learn入門
3. モデルとは？（問題設定、アルゴリズム、モデル）
4. 線形回帰モデル
5. 仮説、損失関数、目的関数
6. 最小二乗法
7. 最急降下法
8. 参考サイト



機械学習は教師あり学習、教師なし学習、強化学習に大別されることを既に述べた。
ここからはブラックボックスとして使っていた「モデル」について中身を見ていこう。

Models

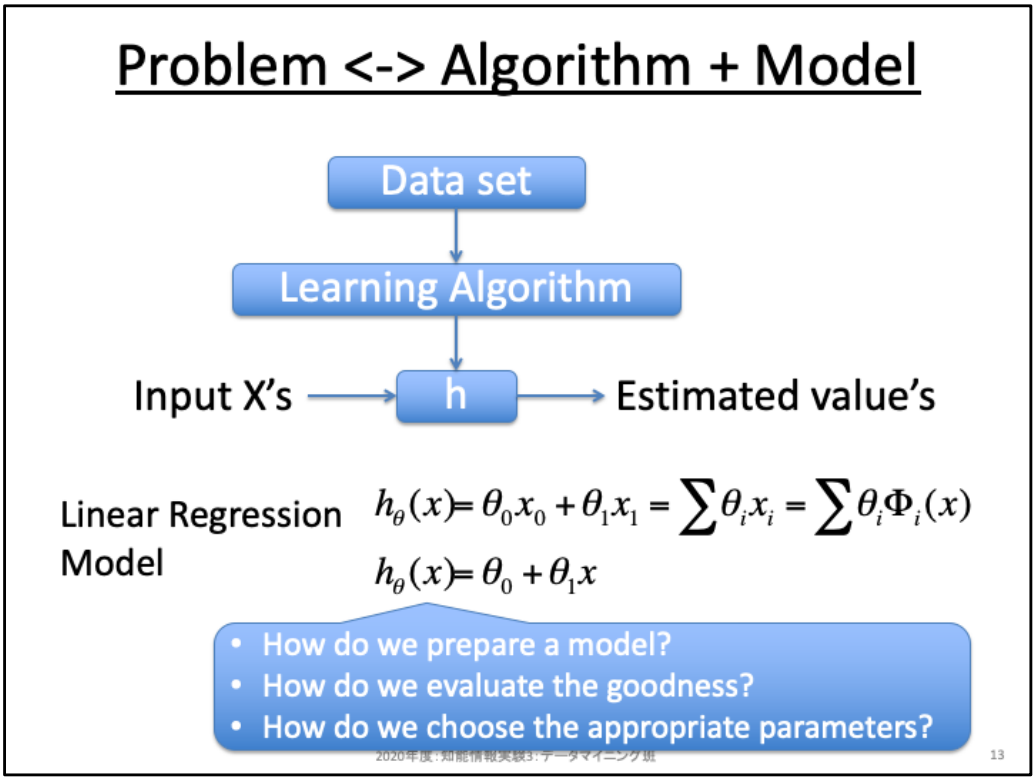
- Represent by any formulas with (sometimes one) parameters for the relationship between input X's and output Y's.
  - In machine learning, the formulas called as "hypothesis".
  - E.g., h = a*x + b
    - a, b: parameters
  - Parameterized model.
  - Predictive model. (e.g., a=1, b=2)



用意したデータセットにおける入出力関係をうまく表現できるようなモデルを用意し、そのモデルにおける適切なパラメータを学習アルゴリズムにより獲得することを目指す。用意したサンプルから、その事象に関する入出力関係をうまく表現できるモデルを獲得できたならば、その学習済みモデルは未知のサンプルに対しても適切な予測結果を出力できる（だろう）。

例えば、h(x) = ax + b というモデルを考えたとしよう。このままでは何も予測できないが、パラメータであるaとbに任意の実数を与えてやると、予測可能なモデルになる。

あるモデルAに対し、異なる学習アルゴリズムがN個あることも珍しくないし、その逆に、ある学習アルゴリズムZを、異なるモデルへ適用することもある。



機械学習を適用する流れとしては、(1)データセットを用意し、(2)モデルを選択し、(3)モデルにデータセットを適用することでそのデータセットをうまく説明するようなパラメータを学習することになる。モデルは、前ページにおけるパラメータ付き数式h(x)と、パラメータをどのように調整するかを定めた学習アルゴリズムから成り立つ。

線形回帰モデルの例では、sklearn.linear_model.LinearRegressionを利用しており、アルゴリズムは plain Ordinary Least Squares (scipy.linalg.lstsq, 最小二乗法) が使われている。線形回帰モデルをより一般化すると、入力xは複数項目（複数特徴量）あって良く、各入力に応じてその重要度を調整するためのパラメータθが用意される。また、入力xに依存しないバイアス項としてθ0も用意する。上記スライドのx0は、ベクトル（もしくは行列）演算として一般化するために用意された定数項（x0=1）である。また、入力xをそのまま扱うのではなく、前処理を加えた値を用いることもある（φ）。

情報工学実験4：データマイニング班

(week 3) 線形回帰モデルと最急降下法

1. 復習
2. scikit-learn入門
3. モデルとは？（問題設定、アルゴリズム、モデル）
4. 線形回帰モデル
5. 仮説、損失関数、目的関数
6. 最小二乗法
7. 最急降下法
8. 参考サイト

Linear Regression Model

- Training datasets
  - (x,y) = (4,7), (8,10), (13,11), (17,14)
- Hypothesis

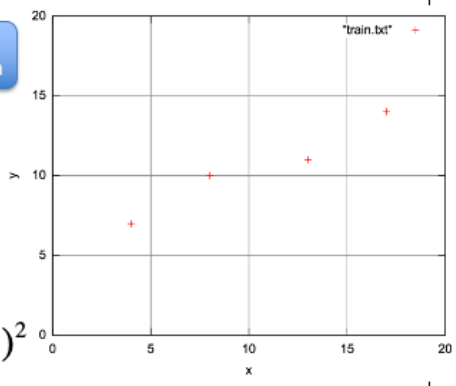
  Assumption 1 Linear function

$$h_\theta(x) = \theta_0 + \theta_1 x$$

- Parameters
  - θ_0, θ_1
- Cost function

  Assumption 2 Squared error

$$J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$



- Objective function (measurement of the goodness)

$$\min_\theta J(\theta_0,\theta_1)$$

2020年度：知能情報実験3：データマイニング班 15

線形回帰モデルの中身を確認していこう。考えやすくするため、4つのサンプルを用意した。目標はこの4つのサンプルをうまく説明するモデルを獲得することである。

まず線形回帰モデルにおいては「モデルは線形で近似する」という前提で考える。ここで事象を近似するとは、h(x)という線形モデルを踏まえ、全てのデータセットに対して近い出力をすることができるパラメータθを求める、ということである。近似するパラメータを求めることができれば、データセットとしては用意しなかった未知の入力xに対しても適切な出力yを予測することができるだろう。一方で、そもそも事象が二次関数である等の理由から直線では近似できない場合、線形モデルでは妥当なモデルを構築することが困難である。しかしながらそもそも事象が「線形」なのかどうなのかは、必ずしも明瞭ではないし、そもそも明瞭であるなら最初からそれに適したモデルを選択するべきである。このような理由から、機械学習においてはモデルを「仮説（hypothesis）」と呼び、その仮説を前提とした最適なパラメータを求めようとする。

パラメータ付き仮説を用意したら、その次に考えるべきことは「あるパラメータを与えたときに、そのモデルの質をどう評価するか」という評価指標についてである。これが赤字で示した損失関数（cost function）や目的関数（objective function）である。損失関数は、パラメータθに基づいた予測結果h(x)と実際の教師データyとの差分の2乗を求め、その総和で算出する。これを二乗和誤差（sum of squared errors of prediction）や残差平方和（residual sum of squares, RSS）と呼ぶ。この値が小さいほど良質なパラメータであと判断できるため、損失関数と呼ぶ。損失関数が最小となるパラメータθを探すことが目的であるため、min J(θ)を目的関数と呼ぶ。

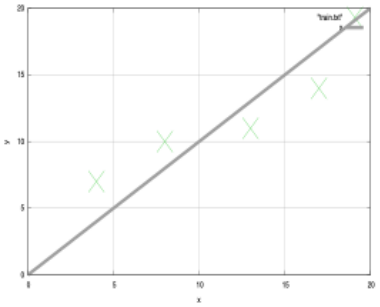
これらの仮説、損失関数、目的関数により「モデル」が構成されている。入出力を固定したままでより良いモデルを作り上げるためには、これらの構成要素を検討することになる。例えば損失関数としてRSS以外にはどのような計測方法が考えられるだろうか？　残差平方和との違いはどうだろうか？

Hypothesis vs. Cost function (θ1=1)

θ0=0, θ1=1, (x,y)=(4,7), (8,10), (13,11)

Hypothesis:

$$h(x) = x$$

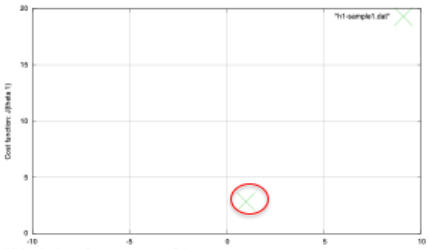


Cost function:

$$J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J(0,1) = \frac{1}{2m}((4-7)^2 + (8-10)^2 + (13-11)^2)$$

$$J(0,1) = \frac{1}{2*3}(9+4+4) = \frac{17}{6} = 2.83$$



コスト関数の値を実際に求めてみよう。ここでは流れを確認することが目的のため、パラメータθ0=0で固定とし、θ=1のとき、すなわちh(x)=xというモデルを考えてみよう。また、サンプル数も3つだけにした。h(x)=xを可視化したのが左図である。このように一度パラメータを与えてしまえば、データセットとしては与えなかったxに対しても出力を予測することができる。例えばx=0.1ならば予測結果h(x)=0.1である。

h(x)=xにおけるコスト関数Jはスライド右の通りであり、コストは2.83となる。機械学習ではよりコストが少なくなるようなパラメータθを探すことが目的であることから、このことを目視しやすくするため、パラメータθを横軸、その時のコストを縦軸としてプロットした図が右下の図である。コストが小さくなるパラメータθを探すということは、この図における横軸を移動しながら縦軸が0に近くなる場所を探していくことに相当する。

Hypothesis vs. Cost function (θ1=0.5)

θ0=0, θ1=0.5, (x,y)=(4,7), (8,10), (13,11)

Hypothesis:

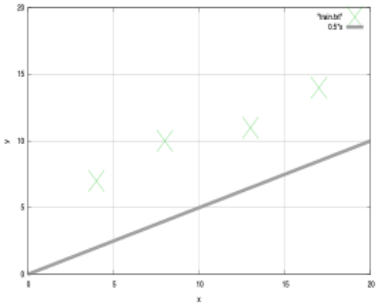
$$h(x) = 0.5 * x$$

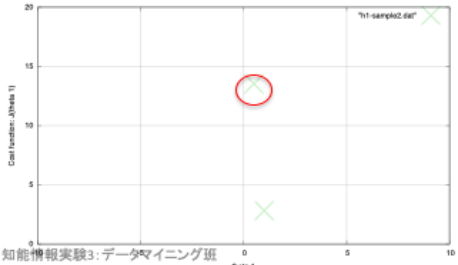
Cost function:

$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J(0.5) = \frac{1}{2m}((2-7)^2 + (4-10)^2 + (6.5-11)^2)$$

$$J(0.5) = \frac{1}{2*3}(25 + 36 + 20.25) = \frac{81.25}{6} = 13.54$$





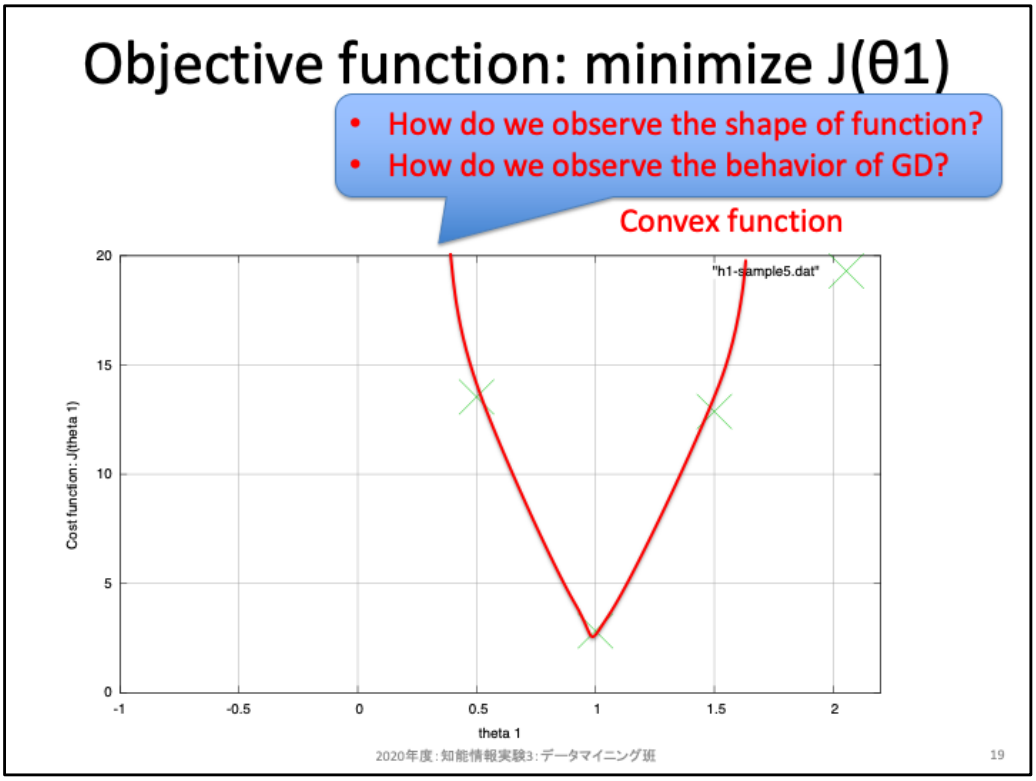
パラメータθ1を少し移動させ、0.5時のコスト関数Jを求めてみた様子を示している。左図はモデルを示しており、全てのサンプルに対して下回る予測値を出力するようになってしまっている。また右下の図から分かるように、このスケールではほんの少し左に移動しただけでもコストが大きく跳ね上がっている（ようにみえる）ことが分かる。なんとなくだが、θ1をこの方向（0.5からみて左方向）に移動するのはよろしく無さそうだ。このような判断を主観ではなく客観的にするために損失関数を定義し、評価指標として用いている。

Hypothesis vs. Cost function (θ1=others)

θ0=0, θ1=others, (x,y)=(4,7), (8,10), (13,11)

- θ1=0:
  - H(x)=0*x=0
  - J(0)=1/6{(0-4)^2+(0-10)^2+(0-13)^2}
    - =1/6{16+100+169}=47.5
- θ1=2:
  - H(x)=2*x
  - J(2)=1/6{(8-7)^2+(16-10)^2+(26-11)^2}
    - =1/6{1+25+225}=41.83
- θ1=1.5:
  - H(x)=1.5*x
  - J(1.5)=1/6{(6-7)^2+(12-10)^2+(19.5-11)^2}
    - =1/6{1+4+72.25}=12.87

パラメータθ1を0, 2, 1.5に移動させたときのコストを計算してみた。これを描画すると次のスライドのとおりとなる。



このように、パラメータθ1だけを調整した場合には1のときが恐らく最もコストが小さくなりそうだ。コスト関数は二乗和誤差になっていることからその関数自体は二次関数である。二次関数ということは上もしくは下に凸となる関数だ。二条和誤差に関して言えばマイナスを取ることがありえないため、下に凸となる。このときの最小値とは傾きが0となる場所であり、理論的には微分（偏微分）や最小二乗法により最適な値を求めることが可能だ。

情報工学実験4：データマイニング班

(week 3) 線形回帰モデルと最急降下法

1. 復習
2. scikit-learn入門
3. モデルとは？（問題設定、アルゴリズム、モデル）
4. 線形回帰モデル
5. 仮説、損失関数、目的関数
6. 最小二乗法
7. 最急降下法
8. 参考サイト

Ordinary Least Squares

problems?

$h(x) = \theta_0 + \theta_1 x$ (x,y)=(4,7), (8,10), (13,11), (17,14)

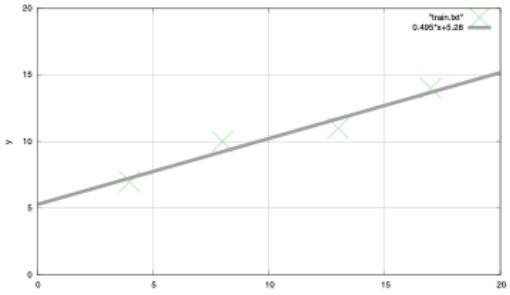
$7 = \theta_0 + 4\theta_1$

$0 = \theta_0 + 4\theta_1 - 7$

$e_1 := \theta_0 + 4\theta_1 - 7$

$e_1^2 = (\theta_0 + 4\theta_1 - 7)^2$

$E = \sum e_i^2 \geq 0$



$= (\theta_0 + 4\theta_1 - 7)^2 + (\theta_0 + 8\theta_1 - 10)^2 + (\theta_0 + 13\theta_1 - 11)^2 + (\theta_0 + 17\theta_1 - 14)^2$

$= 538\theta_1^2 + 84\theta_0\theta_1 + 4\theta_0^2 - 978\theta_1 - 84\theta_0 + 466$

$= (2\theta_1 + 21\theta_0 - 21)^2 + 97(\theta_0 - 48/97)^2 + 121/97$

θ0=1029/194≒5.28, θ1=48/97≒0.495

h(x) = 5.28 + 0.495x

Ref., http://gihyo.jp/dev/serial/01/machine-learning/0008

最小二乗法により求めたθを用いると、右図のように全てのサンプルをうまく近似するモデルとなっていることが分かる。このように、最小二乗法は最適な近似解を求めることができるが、その一方では計算リソースの問題がある。サンプル数が大きすぎたり、次元数（説明変数）が大きすぎると、実用的な時間では解を求めることが困難となる。どのぐらいから困難なのかは計算機と実装に依存するため、実験してみよう。

大規模なデータセットの場合には実用的ではないため、代替案として数値微分（偏微分）に基づく最急降下法を用いることが多い。これは多くのアルゴリズムのコアであることも少なくないため、最急降下法について学んでいこう。

情報工学実験4：データマイニング班
(week 3) 線形回帰モデルと最急降下法

1. 復習
2. scikit-learn入門
3. モデルとは？（問題設定、アルゴリズム、モデル）
4. 線形回帰モデル
5. 仮説、損失関数、目的関数
6. 最小二乗法
7. 最急降下法
8. 参考サイト

Gradient descent algorithm

Repeat until convergence {

$$\theta_i := \theta_i - \alpha \frac{\partial}{\partial \theta_i} J(\theta_0, \theta_1)$$

}

(1) Start with any parameters.
(2) Update the parameters simultaneously, until convergence.

Simple example

$$J(\theta) = \theta^2, \alpha = 0.1$$

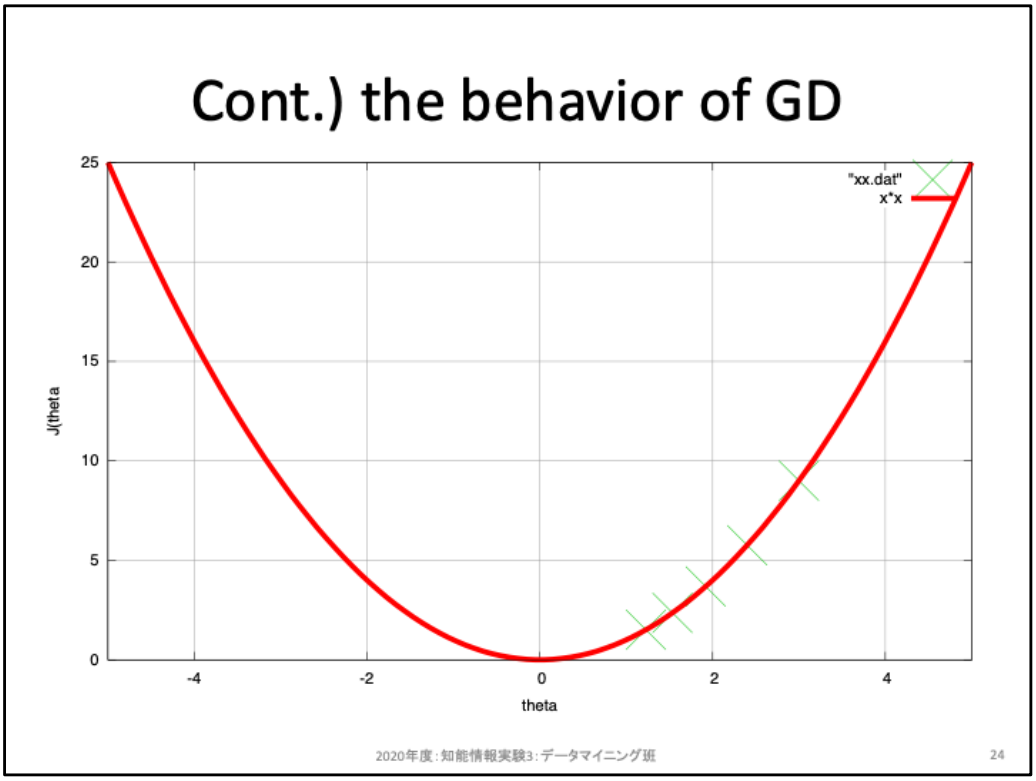
$$new\_\theta = \theta - \alpha \frac{d}{d\theta} J(\theta)$$
$$= \theta - 0.1 * 2\theta = \theta - 0.2\theta = 0.8\theta$$

- e.g., α=0. 1, θ=3, J(θ)=9
- 1st update
  - New_θ = 0.8*3 = 2.4
  - J(θ) = 2.4**2 = 5.76
- 2nd update
  - New_θ = 0.8*2.4 = 1.92
  - J(θ) = 1.92**2 = 3.6864
- 3rd update
  - New_θ = 1.536
  - J(θ) = 2.359296
- 4th update
  - New_θ = 1.2288000000000001
  - J(θ) = 1.5099494400000002

最急降下法のアルゴリズムはとてもシンプルかつ強力だ。そのため様々なアルゴリズムに応用されている。まず、(1)パラメータを任意の値で初期化しよう。その後はスライド上部の式に基づき、(2)収束するまでパラメータを更新し続けよう。パラメータ更新は「同時」である点に注意しよう。例えば2つのパラメータがある状況において、下記ケース1はパラメータθ1を更新する前にθ0が書き換わっているため、不適切である。ケース2のようにコスト算出と更新を分けて処理しよう。

ケース1
θ0におけるコスト算出。
θ0を更新。
θ1におけるコスト算出。
θ1を更新。
ケース2
θ0におけるコスト算出。
θ1におけるコスト算出。
θ0を更新。
θ1を更新。

更新式は2つの項目で構成されている。1つ目の項は現時点でのパラメータそのものである。2つ目の項が「現時点でのパラメータを基準として、どのように変更するのか」を記述している項だ。αは学習係数と呼ばれる項であり、詳細は後述する。残された項は「-偏微分」である。偏微分は指定されたパラメータにおける傾きを求めており、例えばθ0を指定されたならθ1を定数項として扱い傾きを求める。このとき、傾きの大きさは無視し、正負についてのみ考えてみよう。コスト関数は下向きに凸の関数であることを既に述べた。この状況下で、傾きが正（もしくは負）であるとはどのような状況にあるだろうか。傾きが正ならば、そのパラメータを右に移動するとコストは右上に、つまりコストが増えていくはずだ。逆に言えば、傾きが正のときには左方向を探すことでコストが低くなるだろう。同様に傾きが負のときには、より小さなコストはパラメータを右側に移動したほうが良い。これを表しているのが「-」であり、傾きの正負に対して逆方向にパラメータを更新することを意味している。また、αは移動する際の移動幅を調整するためのハイパーパラメータである。αが大きいほど一度に移動する幅が大きくなり、更新回数が少なくて済むかもしれない。しかしながら大きすぎると傾きが0の点には近づけない（移動幅が大きいため飛び越してしまう）可能性も大きくなるため、一般的には小さな値を指定する。



前のスライドにおける右側では、「α=0. 1, θ=3, J(θ)=9」におけるパラメータ行進の様子を示していた。これを図示したのがこのグラフである。最初は右側の×から出発し、少しずつ左側に移動している様子が分かるだろう。同時に、αを固定しているにも関わらず移動幅が少しずつ小さくなっていることも観察できる。これは更新式がαと傾きの積になっているためであり、移動する都度傾きが小さくなるため、結果として移動幅も小さくなっていてる。このためなだらかな傾きが続くような状況下では、なかなか収束しないことが多い。

Gradient descent for Linear Regression

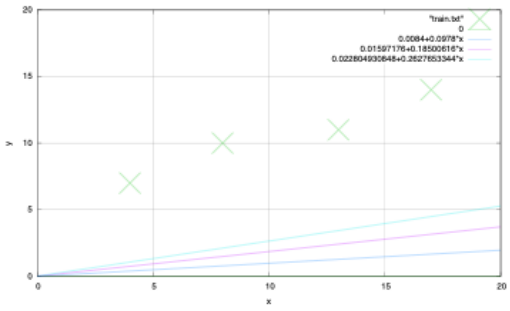
$$\theta_i := \theta_i - \alpha \frac{\partial}{\partial \theta_i} J(\theta_0, \theta_1)$$

$h_\theta(x) = \theta_0 + \theta_1 x$ (x,y) = (4,7), (8,10), (13,11), (17,14)

$$J(\theta_0, \theta_1) = 538\theta_1^2 + 84\theta_0\theta_1 + 4\theta_0^2 - 978\theta_1 - 84\theta_0 + 466$$

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = 84\theta_1 + 8\theta_0 - 84$$

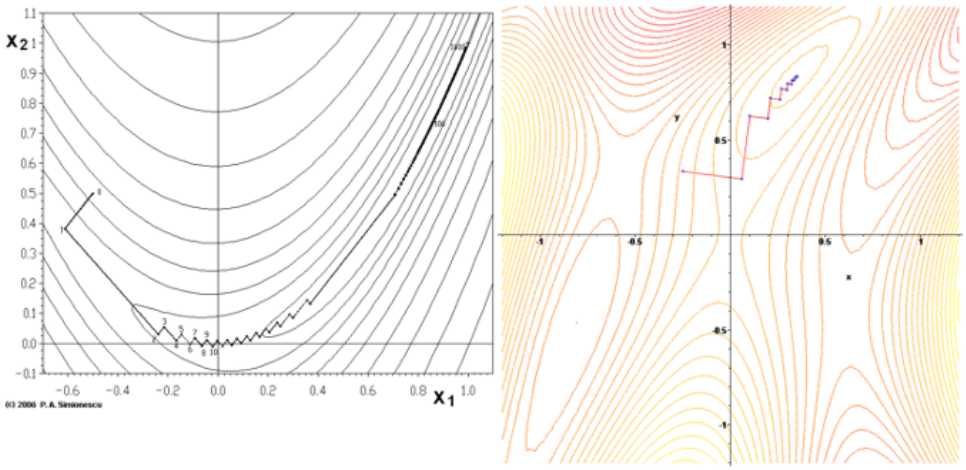
$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = 1076\theta_1 + 84\theta_0 - 978$$



- e.g., α=0.001, θ0=0, θ1=0, J(θ)=466
- 1st update
  - New_θ0 = 0 – 0.001*(-84) = 0.0084
  - New_θ1 = 0 – 0.001*(-978) = 0.0978
  - J(θ) = 374.86117384
- 2nd update
  - New_θ0 = 0.01597176
  - New_θ1 = 0.18500616
  - J(θ) = 302.3858537133122
- 3rd update
  - New_θ0 = 0.022804930848
  - New_θ1 = 0.2627653344
  - J(θ) = 244.75187010633334
- 4th update
  - New_θ0 = 0.0289794580943616
  - New_θ1 = 0.3321002229994368
  - J(θ) = 198.91981002677187

振り返りを兼ねて、パラメータを更新する様子をコスト関数ではなくモデルで可視化した様子を示している。最初は傾きθ1=1から始まっているが、パラメータを更新するたびに傾きが少しずつ増えていっていることが分かる。またバイアス項θ0も徐々に増えていっている。

(optional) zig-zagging behavior



Ref., http://en.wikipedia.org/wiki/Gradient_descent

最急降下法は、そのままではジグザグに移動する。実際に可視化してみるといいだろう。このような移動を無駄とみなしてスムーズに移動する方法もいろいろと提案されているが、本実験ではここまでとする。興味がある人は「**ゼロから作るDeep Learning ―Pythonで学ぶディープラーニングの理論と実装**」で勉強してみよう。

References

- Machine Learning | Coursera, https://class.coursera.org/ml-007
- Gradient descent – Wikipedia, http://en.wikipedia.org/wiki/Gradient_descent
- 数理計画法 第12回, http://www.dais.is.tohoku.ac.jp/~shioura/teaching/mp11/mp11-12.pdf
- 機械学習 はじめよう 第8回　線形回帰[前編], http://gihyo.jp/dev/serial/01/machine-learning/0008
- 機械学習 はじめよう 第9回　線形回帰[後編], http://gihyo.jp/dev/serial/01/machine-learning/0009
- PRMLの線形回帰モデル(線形基底関数モデル), http://www.slideshare.net/yasunoriozaki12/prml-29439402
- An introduction to machine learning with scikit-learn, http://scikit-learn.org/stable/tutorial/basic/tutorial.html