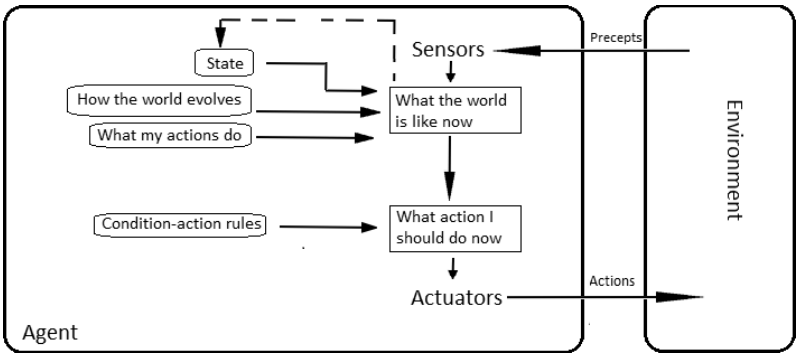
Artificial Intelligence
Kuliah 2: Regresi Linear

Ali Akbar Septiandri
Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Al Azhar Indonesia
aliakbars@live.com

1 Model-based Reflex Agents

Seperti yang sudah dibahas di pertemuan sebelumnya, sebuah agen cerdas akan terdiri dari model yang ditambahkan kemampuan tertentu. Kali ini, kita akan berfokus pada agen dengan model yang ditambahkan kemampuan refleks. Refleks dalam kasus ini berarti bahwa model yang digunakan hanya dapat bereaksi pada satu waktu tertentu. Untuk membangun refleks tersebut, agen dapat **belajar** dengan menggunakan data latih $\mathcal{D}_{train}$ untuk menghasilkan fungsi prediksi f yang akan menerima masukan x dan memetakannya ke $y = f(x)$. Arsitektur dari agen cerdas tipe ini dapat dilihat pada Gambar 1.



Gambar 1: Agen yang mempunyai refleks berbasis model

Penting bagi agen cerdas tersebut untuk menghasilkan fungsi prediksi yang dapat bekerja bahkan untuk data yang belum dilihat dalam $\mathcal{D}_{train}$. Hal ini yang disebut sebagai generalisasi dan akan dibahas pada pertemuan kelima. Namun, secara umum, dalam tugas yang dikenal sebagai ***supervised learning*** ini, kita akan menerima pasangan (x, y) yang menspesifikasikan y sebagai keluaran yang tepat dari masukan x. Oleh karena itu, model yang dihasilkan dapat dievaluasi kinerjanya. Dalam gambar di atas, y dapat diartikan sebagai **"aksi"** yang diprediksi berdasarkan x yang **diekstraksi dari persepsi**.

Agen tipe ini tergolong inteligensi yang rendah karena memerlukan pasangan (x, y) tadi untuk mengevaluasi modelnya. Padahal, kenyataan di lapangan menunjukkan bahwa sulit sekali untuk mendapatkan **label** atau **kelas** y tersebut. Ditambah lagi, "aksi" tersebut tidak selalu berupa aksi yang memengaruhi lingkungan. Kita hanya mendapatkan prediksi $f(x)$ yang dapat dievaluasi dengan nilai (label; y) sebenarnya. Seiring dengan bertambahnya data latih, harapannya model kita pun akan menghasilkan prediksi yang lebih baik.

2 Model Linear

Dalam kasus **regresi**, nilai y yang digunakan bersifat **kontinu**. Di sisi lain, dalam kasus **klasifikasi**, nilai y yang digunakan bersifat **diskrit**. Kali ini, kita akan berfokus pada kasus regresi dan bagaimana menghasilkan model yang dapat mengerjakan tugas tersebut. Materi tentang klasifikasi akan diulas pada pertemuan berikutnya.

2.1 Ordinary Least Squares

Dalam bentuk yang paling sederhana, kita dapat mencocokkan garis lurus ke $\mathcal{D}_{train}$ yang mengikuti fungsi:

$$y = w_0 + w_1 x_1$$

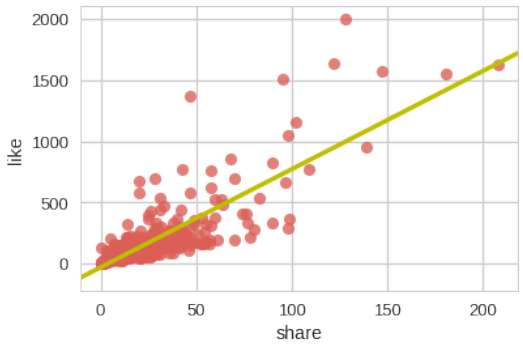
dengan w_0 disebut juga sebagai bias (terkadang ditulis juga dalam simbol b) dan w_1 adalah *slope* atau gradien yang menentukan kemiringan garis lurus tersebut. Metode *ordinary least squares* mencoba mencari garis yang menghasilkan nilai kuadrat error terkecil atau

$$\arg\min_{w_0,w_1} \frac{1}{|\mathcal{D}_{train}|} \sum_{(x,y)\in\mathcal{D}_{train}} \ell(y, x, w_0, w_1)$$

dengan nilai $\ell(y, x, w_0, w_1)$ dapat dirumuskan sebagai

$$\{y - (w_0 + w_1 x)\}^2$$

Sebagai ilustrasi, garis berwarna kuning pada Gambar 2 adalah fungsi linear yang meminimalkan rata-rata error atau residu dari semua titik merah. Dalam kasus tersebut, x adalah jumlah *share* dan y adalah jumlah *like*. Bagaimana jika kita ingin memetakan lebih banyak masukan?



Gambar 2: Hasil *least squares regression*

2.2 Regresi Linear Multidimensi

Contoh kasus Anda diberikan sekumpulan foto dari Facebook. Tugas Anda adalah membuat sebuah *regressor* yang dapat memprediksi jumlah *like* yang akan didapatkan foto tersebut jika diberikan gambarnya dan beberapa statistiknya, e.g. jumlah *share*, jumlah komentar, dan profil pengepos. Bagaimana cara Anda dapat menghasilkan model yang dapat melakukan prediksi tersebut?

Hal pertama yang harus dilakukan adalah mengekstraksi fitur dari gambar atau statistik yang diberikan. Untuk mempermudah penjelasan saat ini, maka kita akan menggunakan fitur dari statistik foto tersebut saja.

Definisi Untuk masukan x, vektor fiturnya adalah:

$$\phi(x) = [\phi_1(x), ..., \phi_d(x)].$$

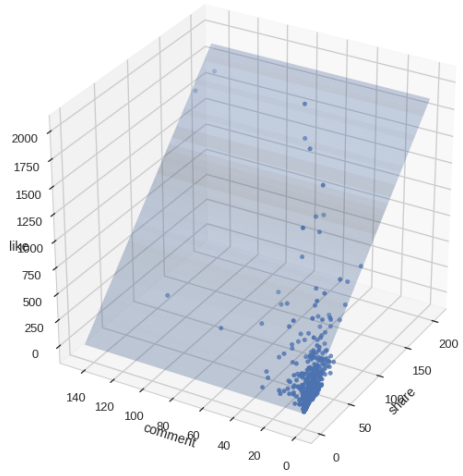
Kita dapat melihat $\phi(x) \in \mathbb{R}^d$ sebagai sebuah titik di ruang berdimensi tinggi. Dengan demikian, fungsi regresi linear untuk multidimensi menjadi

$$y = w_0 x_0 + w_1 x_1 + w_2 x_2 + ... + w_D x_D = \sum_{j=0}^{D} w_j x_j = \mathbf{w} \cdot \phi(x)$$

Sebagai contoh, fitur dari statistik sebuah foto dapat direpresentasikan dalam vektor:

share	:	49
comment	:	19
paid	:	0

Seandainya kita hanya menggunakan dua fitur: *share* dan *comment*, maka yang dihasilkan oleh *least squares regression* akan berbentuk bidang alih-alih garis seperti dapat dilihat pada Gambar 3. Untuk dimensi yang lebih tinggi, hasilnya akan berbentuk *hyperplane*. Pertanyaannya, bagaimana mendapatkan nilai $\mathbf{w}$ yang akan membentuk *hyperplane* ini?



Gambar 3: Hasil *least squares regression* dengan $\phi(x) \in \mathbb{R}^2$

3 Meminimalkan Error

Seperti yang disinggung saat membahas *ordinary least squares* di atas, tujuan kita adalah menghasilkan nilai kuadrat error terkecil yang dapat didefinisikan sebagai

$$E(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \mathbf{w} \cdot \phi(x_i))^2$$

dengan N adalah jumlah data yang ada pada $\mathcal{D}_{train}$. Perhatikan bahwa indeks i di sini merujuk kepada data ke-i, sedangkan indeks j di atas merujuk pada dimensi ke-j.

Nilai $E(\mathbf{w})$ dapat diminimalkan dengan mencari turunan pertama, lalu mengatur nilainya menjadi 0, i.e. $\nabla_{\mathbf{w}} E(\mathbf{w}) = 0$.[1] Persamaan ini akan menghasilkan solusi tertutup[2]

$$\hat{\mathbf{w}} = (\boldsymbol{\phi}^T\boldsymbol{\phi})^{-1}\boldsymbol{\phi}^T\mathbf{y}$$

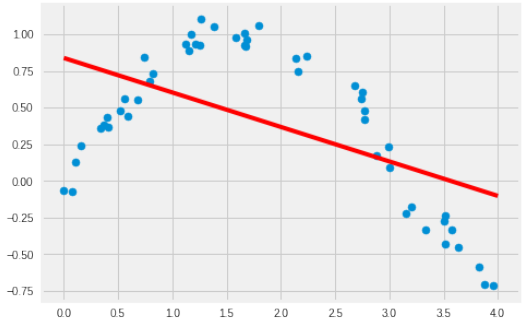
Perhatikan bahwa persamaan tersebut menggunakan $\boldsymbol{\phi}$ dan $\mathbf{y}$ yang berarti matriks dan vektor, secara berturut-turut. Bagian $(\boldsymbol{\phi}^T\boldsymbol{\phi})^{-1}\boldsymbol{\phi}^T$ dikenal sebagai *pseudo-inverse* karena tidak mungkin[3] untuk mencari langsung nilai invers dari $\boldsymbol{\phi}$.

4 Fungsi Non-linear

Karena tidak semua tren dalam data berupa hubungan linear, terkadang kita perlu melakukan transformasi non-linear terhadap data sehingga dapat menghasilkan model yang dapat menangani hal

[1] Ingat kembali cara mencari titik puncak dari sebuah fungsi pada saat pelajaran kalkulus dulu.
[2] Silakan buktikan persamaan ini sebagai latihan lebih lanjut.
[3] Mengapa?

tersebut. Sebagai contoh, Gambar 4 menunjukkan hasil *least squares regression* tanpa transformasi non-linear. Kita dapat melihat bahwa trennya menunjukkan hanya ada *satu titik puncak*. Maka, kita dapat mencoba untuk melakukan transformasi dengan menggunakan **fungsi basis polinomial** berpangkat dua.

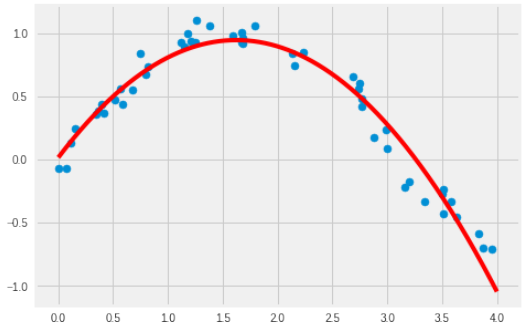


Gambar 4: Hasil *least squares regression* dari data yang dihasilkan dengan fungsi $\sin$ dengan *noise*

Definisi Jika kita mengubah $x_p = f_p(x)$, dengan $f_p()$ adalah fungsi transformasi, maka untuk $f_p() = x^p$ dan x adalah input berdimensi satu, modelnya menjadi

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + ... + w_p x^p$$

Berdasarkan definisi di atas, maka kita dapat membuat *regressor* seperti pada Gambar 5 dari hasil transformasi dengan fungsi basis polinomial. Dapat dilihat bahwa tidak semua titiknya berada pada garis merah. Hal ini dilakukan karena kita ingin melihat tren utamanya dan "menerima" *noise* yang ada untuk menghasilkan model yang generik.



Gambar 5: Hasil *least squares regression* dari data setelah transformasi dengan fungsi basis polinomial

Dengan menambahkan pangkat pada fungsi basis polinomial, maka *regressor* yang dihasilkan akan lebih fleksibel karena parameternya ($\mathbf{w}$) pun menjadi semakin banyak. Akan tetapi, model ini akan mengalami ***overfitting*** sehingga tidak akan bekerja baik jika diberikan data baru yang belum pernah dilihat sebelumnya. Jadi, penting bagi kita untuk "mendistilasi" tren utama dari data dengan mengecilkan pengaruh *noise* dalam pembentukan model.

Anda dapat menganalogikan kasus *overfitting* seperti seorang mahasiswa yang hanya belajar berdasarkan soal ujian tahun lalu saja untuk suatu mata kuliah. Mahasiswa tersebut ternyata mendapatkan nilai yang jelek pada ujian karena soal yang diberikan berbeda dari soal ujian tahun lalu meski masih dalam satu topik yang sama. Lebih lanjut tentang kasus *overfitting* dan *underfitting* akan dibahas pada pertemuan kelima.

References

[1] VanderPlas, J., 2016. *Python Data Science Handbook*. O'Reilly Media Inc.