Feature Selection and Generation

Peng Lingwei

August 23, 2019

Contents

25 Feature Selection and Generation

1. Feature selection: we have a large pool of features and our goal is to select a small number of features that will be used by predictor;
2. Feature manipulations, normalization: decrease the sample complexity of our learning algorithm, bias or computational complexity.
3. Feature learning.
4. note: the No-Free-Lunch theorem implies that there is no ultimate feature learner. Any feature learning algorithm might fail on some problem. The success of each feature learner relies on some form of prior assumption on the data distribution, and depends on the learning algorithm that uses these features.

25.1 FEATURE SELECTION

25.1.1 Filters

Filters: score individual features, and choose k features that achieve the highest score.

Let $X = [\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_m] = [\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_n]^T$. Consider a linear regression problem, we score these features by using individual empirical squared loss.

$$Score(\vec{v}) = \min_{a,b \in \mathbb{R}} \frac{1}{m} \|a\vec{v} + b - \vec{y}\|^2$$

We can simplify this score.

First, let $\bar{v} = \frac{1}{m}\sum_{i=1}^m v_i$, and $\bar{y} = \frac{1}{m}\sum_{i=1}^m y_i$, then

$$Score(\vec{v}) = \min_{a,b \in \mathbb{R}} \frac{1}{m} \|a\vec{v} + b - \vec{y}\|^2 = \min_{a,b \in \mathbb{R}} \frac{1}{m} \|a(\vec{v} - \bar{v}) + b - (\vec{y} - \bar{y})\|^2$$

Proof. Let a^*, b^* satisfy $\min_{a,b} \in \mathbb{R}\frac{1}{m}\|a\vec{v} + b - \vec{y}\|^2$, then let $a' = a^*, b' = b^* + a^*\bar{v} - \bar{y}$, then

$$\frac{1}{m}\|a'(\vec{v} - \bar{v}) + b' - (\vec{y} - \bar{y})\|^2 = \frac{1}{m}\|a^*\vec{v} + b^* - \vec{y}\|$$

which implies that

$$\min_{a,b \in \mathbb{R}} \frac{1}{m} \|a\vec{v} + b - \vec{y}\|^2 \geq \min_{a,b \in \mathbb{R}} \frac{1}{m} \|a(\vec{v} - \bar{v}) + b - (\vec{y} - \bar{y})\|^2$$

Doing the same for the other direction, we get the target equation. □

When we solve the problem $\min_{a,b \in \mathbb{R}} \frac{1}{m}\|a(\vec{v} - \bar{v}) + b - (\vec{y} - \bar{y})\|^2$, we can get $b = 0$, $a = \langle \vec{v} - \bar{v}, \vec{y} - \bar{y} \rangle / \|\vec{v} - \bar{v}\|^2$, then

$$Score(\vec{v}) = \|\vec{y} - \bar{y}\|^2 - \frac{(\langle \vec{v} - \bar{v}, \vec{y} - \bar{y} \rangle)^2}{\|\vec{v} - \bar{v}\|^2}$$

So we can define a new score function

$$Score(\vec{v}) = \frac{\frac{1}{m}\langle \vec{v} - \bar{v}, \vec{y} - \bar{y} \rangle}{\sqrt{\frac{1}{m}\|\vec{v} - \bar{v}\|^2}\sqrt{\frac{1}{m}\|\vec{y} - \bar{y}\|^2}}$$

The preceeding expression is know as Pearson's correlation coeffcient.

Note: If $Score(\vec{v}) = 0$ means that the optimal linear function from $\vec{v}$ to $\vec{y}$ is the all-zeors function, which means that $\vec{v}$ alone is useless for predicting $\vec{y}$. However, this does not mean that $\vec{v}$ is a bad feature, as it might be the case that together with other features $\vec{v}$ can perfectly predict $\vec{y}$.

25.1.2 Greedy Selection Approaches

Forward greedy selection: start with an empty set of features, and then we gradually add one feature at a time to the set of selected features.

$$j_t = \arg\min_{j} \min_{\vec{w} \in \mathbb{R}^t} \|X_{I_{t-1} \cup \{j\}} \vec{w} - \vec{y}\|$$

$$I_t = I_{t-1} \cup \{j_t\}.$$

Example 1. ***(Orthogonal Matching Pursuit).*** *Let V_t be a matrix whose columns form an orthonormal basis of the columns of X_{I_t}, clearly,*

$$\min_{\vec{w}} \|X_{I_t} \vec{w} - \vec{y}\|^2 = \min_{\vec{\theta} \in \mathbb{R}^t} \|V_t \vec{\theta} - \vec{y}\|^2$$

We decompose $X_j = V_{t-1} V_{t-1}^T X_j + \vec{u}_j$, it's easy to verify $\langle \vec{u}_j, V_{t-1}\theta \rangle = 0$.

$$\begin{aligned}
\min_{\vec{w} \in \mathbb{R}^t} \|X_{I_{t-1} \cup \{j\}} \vec{w} - \vec{y}\|^2 &= \min_{\vec{\theta}, \alpha} \|V_{t-1} \vec{\theta} + \alpha \vec{u}_j - \vec{y}\|^2 \\
=& \min_{\vec{\theta}, \alpha} \left[\|V_{t-1}\vec{\theta} - \vec{y}\|^2 + \alpha^2 \|\vec{u}_j\|^2 + 2\alpha \langle \vec{u}_j, V_{t-1}\vec{\theta} - \vec{y} \rangle \right] \\
=& \min_{\vec{\theta}, \alpha} \left[\|V_{t-1}\vec{\theta} - \vec{y}\|^2 + \alpha^2 \|\vec{u}_j\|^2 + 2\alpha \langle \vec{u}_j, -\vec{y} \rangle \right] \\
=& \min_{\vec{\theta}} \left[\|V_{t-1}\vec{\theta} - \vec{y}\|^2 \right] + \min_{\alpha} \left[\alpha^2 \|\vec{u}_j\|^2 + 2\alpha \langle \vec{u}_j, -\vec{y} \rangle \right] \\
=& \|V_{t-1}\vec{\theta}_{t-1} - \vec{y}\|^2 + \min_{\alpha} \left[\alpha^2 \|\vec{u}_j\|^2 + 2\alpha \langle \vec{u}_j, -\vec{y} \rangle \right] \\
=& \|V_{t-1}\vec{\theta}_{t-1} - \vec{y}\|^2 - \frac{(\langle \vec{u}_j, \vec{y} \rangle)^2}{\|\vec{u}_j\|^2}
\end{aligned}$$

So, we should select the feature $j_t = \arg\max_j \frac{(\langle \vec{u}_j, \vec{y} \rangle)^2}{\|\vec{u}_j\|^2}$. The rest of the update is to set

$$V_t = \left[V_{t-1}, \frac{\vec{u}_{j_t}}{\|\vec{u}_{j_t}\|^2} \right], \vec{\theta}_t = \left[\vec{\theta}_{t-1}; \frac{\langle \vec{u}_{j_t}, \vec{y} \rangle}{\|\vec{u}_{j_t}\|}^2 \right]$$

The preceeding procedure is often numerically unstable (Gram-Schmidt procedure).

Algorithm 1 Orthogonal Matching Pursuit (OMP)

```
Require: $X \in \mathbb{R}^{m,d}, \vec{y} \in \mathbb{R}^m$, features num $T$.
Ensure: $I_1 = \emptyset$.
  for $t = 1, \dots, T$ do
      $V = SVD(X_{I_t})$ (If t = 1, V = $\mathbf{0}$).
      for $j \in [d] \backslash I_t$ do
          $\vec{u}_j = X_j - VV^T X_j$
      end for.
      $j_t = \arg\max_{j \in I_t : \|\vec{u}_j\| > 0} \frac{(\langle \vec{u}_j, \vec{y} \rangle)^2}{\|\vec{u}_j\|^2}$
      $I_{t+1} = I_t \cup \{j_t\}$
  end for.
  return $I_{T+1}$
```

More Efficient Greedy Selection Criteria

$$\arg\min_j \min_{\eta \in \mathbb{R}} R(\vec{w}_{t-1} + \eta \vec{e}_j)$$

An even simpler approach is to upper bound $R(\vec{w})$. If R is a $\beta-$smooth function, then

$$\min_{\eta \in \mathbb{R}} R(\vec{w} + \eta \vec{e}_j) \leq \min_{\eta \in \mathbb{R}} R(\vec{w}) + \eta \frac{\partial R(\vec{w})}{\partial w_j} + \beta \eta^2 / 2 = R(\vec{w}) - \frac{1}{2\beta} \left(\frac{\partial R(\vec{w})}{\partial w_j} \right)^2$$

then

$$j_{t+1} = \arg\max_j \left(\frac{\partial R(\vec{w})}{\partial w_j} \right)^2$$

Backward Elimination Another popular greedy selection approach is backward elimination. It is also possible to combine forward and backward greedy steps.

Example 2. ***(AdaBoost as a Forward Greedy Selection Algorithm).***

Proof. $f_{\vec{w}}(\cdot) = \sum_{i=1}^d w_i h_i(\cdot)$, $D_i = \frac{\exp(-y_i f_{\vec{w}}(\vec{x}_i))}{Z}$, where $Z = \sum_{i=1}^m \exp(-y_i f_{\vec{w}}(\vec{x}_i))$.

$$R(\vec{w}) = \log \left(\sum_{i=1}^m \exp\left(-y_i f_{\vec{w}}(\vec{x}_i)\right) \right) = \log \left(\sum_{i=1}^m \exp \left(-y_i \sum_{j=1}^d w_j h_j(\vec{x}_j) \right) \right).$$

$$\begin{aligned} \frac{\partial R(\vec{w})}{\partial w_j} &= -\sum_{i=1}^m D_i y_i h_j(\vec{x}_i) = \sum_{i=1}^m \left\{ D_i 1_{[h_j(\vec{x}_i) \neq y_i]} - D_i 1_{[h_j(\vec{x}_i) = y_i]} \right\} \\ =& 2\sum_{i=1}^m D_i 1_{[h_j(\vec{x}_i) \neq y_i]} - 1 = 2\epsilon_j - 1 \Rightarrow \left| \frac{\partial R(\vec{w})}{\partial w_j} \right| \geq 2\gamma \end{aligned}$$

□

The remaining is analogue in my note of chapter 10.

25.1.3 Sparsity-Inducing Norms

$$\min_{\vec{w}} L_S(\vec{w}) \quad s.t. \quad \|\vec{w}\|_0 \leq k.$$

where

$$\|\vec{w}\|_0 = |\{i : w_i \neq 0\}|$$

The preceeding problem is computationally hard. A possible relaxation is to solve

$$\min_{\vec{w}} L_S(\vec{w}) \quad s.t. \quad \|\vec{w}\|_1 \leq k.$$

In some sense equivalent, the preceeding problem equals to

$$\min_{\vec{w}} L_S(\vec{w}) + \lambda\|\vec{w}\|_1$$

Example 3.

$$\min_{w\in\mathbb{R}} \left(\frac{1}{2}w^2 - xw + \lambda\,|w|\right).$$

Proof.

$$\begin{aligned}
&\min_{w\in\mathbb{R}} \left(\frac{1}{2}w^2 - xw + \lambda\,|w|\right) \\
=&\min\left\{\min_{w>0}\left\{\frac{1}{2}w^2 - (x-\lambda)w\right\}, \min_{w<0}\left\{\frac{1}{2}w^2 - (x+\lambda)w\right\}\right\} \\
=&\begin{cases} \min_{w>0}\left\{\frac{1}{2}w^2 - (x-\lambda)w\right\}, & x \geq \lambda \\ \min_{w<0}\left\{\frac{1}{2}w^2 - (x+\lambda)w\right\}, & x \leq -\lambda \\ 0, & otherwise. \end{cases}
\end{aligned}$$

$$\arg\min_{w\in\mathbb{R}} \left(\frac{1}{2}w^2 - xw + \lambda\,|w|\right) = \begin{cases} x-\lambda, & x \geq \lambda \\ x+\lambda, & x \leq -\lambda \\ 0, & otherwise \end{cases} = sign(x)[|x| - \lambda]_+$$

□

Definition 1. ***(LASSO algorithm).***

$$\arg\min_{\vec{w}} \left(\frac{1}{2m}\|X\vec{w} - \vec{y}\|^2 + \lambda\|\vec{w}\|_1\right)$$

25.2 FEATURE MANIPULATION AND NORMALIZATION

In chapter13, we bound the error with $\|\vec{w}^*\|$. If $\|\vec{w}^*\|$ is large, the sample complexity is large. If we normalize the feature, we can let $\|\vec{w}^*\|^2 = 1$.

If we normalize the feature, it can greatly decrease the runtime of SGD.

Example 4. *In data space, $y \sim U\{\pm 1\}$, $p[x = y|y] = 1 - 1/a$, and $p[x = ay|y] = 1/a$, where $a > 1$.*

$$L_{\mathcal{D}}(w) = \mathbb{E}\frac{1}{2}(wx - y)^2 = \left(1 - \frac{1}{a}\right)\frac{1}{2}(wy - y)^2 + \frac{1}{a}\frac{1}{2}(awy - y)^2$$

Solving for w we obtain that $w^* = \frac{2a-1}{a^2+a-1}$. For $a \to \infty$, $w^* \to \infty$ and $L_{\mathcal{D}}(w^*) \to 0.5$.

If we transform $x \mapsto sign(x) \min\{1, |x|\}$. Then, $w^* = 1$ and $L_D(w^*) = 0$.

Of course, it is not hard to think of examples in which the same feature transformation acturally hurts performance and increases the approximation error.

25.2.1 Examples of Feature Transformations

1. **Centering**: $f_i \leftarrow f_i - \bar{f}$;
2. **Unit Range**: $f_i \leftarrow \frac{f_i - f_{\min}}{f_{\max} - f_{\min}}$ or $f_i \leftarrow 2\frac{f_i - f_{\min}}{f_{\max} - f_{\min}} - 1$;
3. **Standardization**: $\nu = \frac{1}{m}\sum_{i=1}^{m}\left(f_i - \bar{f}\right)^2$ be the empirical variance of the feature, and $f_i \leftarrow \frac{f_i - \bar{f}}{\sqrt{\nu}}$;
4. **Clipping**: $f_i \leftarrow sign(f_i) \max\{b, |f_i|\}$;
5. **Sigmoidal Transformation**: $f_i \leftarrow \frac{1}{1+\exp(bf_i)}$
6. **Logarithmic Transformation**: $f_i \leftarrow \log(b + f_i)$.

25.3 FEATURE LEARNING

Feature learning: learn a function $\psi : \mathcal{X} \to \mathbb{R}^d$.

The No-Free-Lunch theorem tells us that we must incorporate some prior knowledge on the data distribution in order to build a good feature representation.

25.3.1 Dictionary Learning Using Auto-Encoders

1. Encoder function: $\psi : \mathbb{R}^d \to \mathbb{R}^k$;
2. Decoder function: $\phi : \mathbb{R}^k \to \mathbb{R}^d$;
3. Target: $\min_{\phi,\psi}\sum_{i=1}^{m}\|\vec{x}_i - \phi(\psi(\vec{x}_i))\|^2$ with some constraints on ϕ, ψ.

In PAC, we constrain $k < d$ and ψ and ϕ to be linear functions.

In k-means, ψ, ϕ rely on k centroids.ψ returns the index the closest centroid to $\vec{x}$, and ϕ returns the corresponding centroid. On the other words, the k-means ψ returns a vector only a single coordinate of $\psi(\vec{x})$ is nonzero. An immediate extension of the k-means construction is

$$\psi(\vec{x}) = \arg\min_{\vec{v}} \|\vec{x} - \phi(\vec{v})\|^2 \quad s.t. \quad \|\vec{v}\|_0 \leq s.$$

We sometime use l_1 regularization

$$\psi(\vec{x}) = \arg\min_{\vec{v}} \|\vec{x} - \phi(\vec{v})\|^2 + \lambda\|\vec{v}\|_1.$$