

Lecture 3 in computational learning theory

Learning in the $\text{PAC}_U(\text{MQ})$ model using Fourier analysis

Ido Ben Eliezer and Itai Ben Eliezer

In this lecture note we present a method that is used to learn DT in the $\text{PAC}_U(\text{MQ})$ model using Fourier analysis in polynomial time complexity and number of queries. This method was developed in [KM].

Reminder: In the $\text{PAC}_U(\text{MQ})$ model, the learner can get examples $(x, f(x))$ according to the uniform distribution and can ask membership queries $\text{MQ}(a)$. It wants to find with probability at least $1 - \delta$ a function h such that

$$\Pr_U [h(x) \neq f(x)] \leq \epsilon.$$

We begin with introducing a useful theorem in probability.

3.1 Hoeffding - Chernoff Theorem

Theorem 1 (Hoeffding - Chernoff) *Let X_1, \dots, X_m be independent random variables such that $X_i \in [a, b]$ and $E[X_i] = p$ for every i . Then*

$$\Pr \left[\left| \frac{\sum_{i=1}^m X_i}{m} - p \right| > \lambda \right] \leq 2e^{\frac{-2m\lambda^2}{(b-a)^2}}.$$

Remark: From now on, all the probabilities and expectations are taken over the uniform distribution.

Corollary 2 *When flipping an unbiased coin, with possible values $\{0, 1\}$, n times and if S is the sum of the values, then:*

$$\Pr \left[\left| \frac{S}{m} - p \right| > \lambda \right] \leq 2e^{-2m\lambda^2}.$$

Given a hypothesis h , We want to compute $\Pr_D [f(x) \neq h(x)]$. Let

$$m = \frac{1}{2\lambda^2} \ln \left(\frac{2}{\delta} \right)$$

We take m examples $(x_1, f(x_1)), \dots, (x_m, f(x_m))$ according to distribution D . Define a random variable $X_i = 1$ if $h(x_i) = f(x_i)$ and 0 otherwise. Define

$$S = \sum_{i=1}^m X_i.$$

Since

$$E[X_i] = \Pr_D [f(x) \neq h(x)]$$

and

$$2e^{-2m\lambda^2} \leq \delta,$$

by Hoeffding - Chernoff with probability at least $1 - \delta$ we have found a λ -approximation of $\Pr_D [f(x) \neq h(x)]$:

$$\Pr \left[\left| \frac{S}{m} - \Pr_D [f(x) \neq h(x)] \right| < \lambda \right] \geq 1 - \delta. \quad (3.1)$$

Now we use this technique to prove a general theorem in the computational learning theory:

Theorem 3 *If $A(\epsilon, \delta, n)$ is an algorithm that learns C in $PAC_U(MQ)$ model with $\text{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}, n)$ time complexity, then there is algorithm which learns C in $PAC_U(MQ)$ in $\text{poly}(\frac{1}{\epsilon}, \log(\frac{1}{\delta}), n)$ time complexity.*

Proof:

Let A be an algorithm that learns C in $T(\frac{1}{\epsilon}, \frac{1}{\delta}, n)$ time complexity. We create a new algorithm in the following way:

We run $A(\frac{\epsilon}{4}, \frac{1}{2}, n)$, $k = \log(\frac{2}{\delta})$ times. Every time A returns with probability at least $\frac{1}{2}$ a function h such that

$$\Pr[f \neq h] \leq \frac{\epsilon}{4}$$

and thus the probability that for every $1 \leq i \leq k$, $\Pr[f \neq h_i] > \frac{\epsilon}{4}$ is at most $(\frac{1}{2})^{\log(2/\delta)} = \frac{\delta}{2}$. Therefore with probability at least $1 - \frac{\delta}{2}$ there is h_{i_0} such that $\Pr[f \neq h_{i_0}] \leq \frac{\epsilon}{4}$. We use (1) to estimate the value of $\Pr[f \neq h_i]$ using Hoeffding - Chernoff theorem with the parameters $\lambda = \frac{\epsilon}{4}$ and $m = \frac{8}{\epsilon^2} \ln(4k/\delta)$. By (3.1) we can find values q_i such that for every i with probability at least $1 - \frac{\delta}{2k}$

$$|q_i - \Pr[f \neq h_i]| \leq \frac{\epsilon}{4}.$$

Thus with probability at least $1 - \frac{\delta}{2}$ for every i

$$|q_i - \Pr[f \neq h_i]| \leq \frac{\epsilon}{4}. \quad (3.2)$$

In addition, with probability at least $1 - \frac{\delta}{2}$ there is hypothesis h_{i_0} such that

$$\Pr[f \neq h_{i_0}] \leq \frac{\epsilon}{4}. \quad (3.3)$$

Therefore with probability at least $1 - \delta$ both (3.2) and (3.3) happens. Now choose h_{i_1} such that $q_{i_1} \leq \frac{\epsilon}{2}$. since by (3.2)

$$q_{i_0} \leq \frac{\epsilon}{4} + \Pr[f \neq h_{i_0}] \leq \frac{\epsilon}{2}$$

such h_i exists. By (3.2)

$$\Pr[f \neq h_{i_1}] \leq \frac{\epsilon}{2} + q_{i_1} \leq \epsilon$$

with probability at least $1 - \delta$.

It is easy to see that the new algorithm runs in

$$T' \left(\frac{1}{\epsilon}, \frac{1}{\delta}, n \right) = \log \left(\frac{4k}{\delta} \right) T \left(\frac{2}{\epsilon}, 2, n \right) + \frac{8}{\epsilon^2} \log \left(\frac{2}{\delta} \right)$$

time complexity. For

$$T \left(\frac{1}{\epsilon}, \frac{1}{\delta}, n \right) = \text{poly} \left(\frac{1}{\epsilon}, \frac{1}{\delta}, n \right)$$

we have

$$T' \left(\frac{1}{\epsilon}, \frac{1}{\delta}, n \right) = \text{poly} \left(\frac{1}{\epsilon}, \log \left(\frac{1}{\delta} \right), n \right).$$

This completes the proof. ■

3.2 Fourier analysis

From now on, we deal with boolean functions $f : \{0, 1\}^n \rightarrow \{-1, 1\}$. Clearly, if we can learn functions in this form, we can also learn functions in the form $f : \{0, 1\}^n \rightarrow \{0, 1\}$.

Definition 1 Given a set $S \subseteq \{1, \dots, n\}$ we define

$$\chi_S(x) = \prod_{i \in S} (-1)^{x_i}.$$

Define $B = \{\chi_S \mid S \subseteq \{1, \dots, n\}\}$

Remark: It is easy to see that

$$\chi_S(x_1, \dots, x_n) = (-1)^{x_{i_1} \oplus \dots \oplus x_{i_k}}$$

where $S = \{x_{i_1}, \dots, x_{i_k}\}$.

Claim 4 The χ_S functions have the following properties:

(1) $\chi_\emptyset \equiv 1$.

(2) $\chi_S \chi_T = \chi_{S \Delta T}$ when Δ is the symmetric difference between S and T .

(3) If $S = \emptyset$ then $E[\chi_S] = 1$. Otherwise, $E[\chi_S] = 0$.

(4) $\chi_S(x) \chi_S(y) = \chi_S(x + y)$ where $x + y$ is the bitwise XOR of x and y .

Proof:

The proof of (1) is immediate from the definition.

For (2), we know that for every x_i , $((-1)^{x_i})^2 \equiv 1$ and thus the expression

$\chi_S \chi_T$ depends only in the variables that are in the symmetric difference.

For (3), it is clear from the first part of the claim that if $S = \emptyset$ then $E[\chi_S] = 1$. Otherwise, let $\{i\} \subseteq S$ and then

$$\begin{aligned} E[\chi_S] &= E[\chi_{S \setminus \{i\}}(-1)^{x_i}] \\ &= E[\chi_{S \setminus \{i\}}] E[(-1)^{x_i}] = 0. \end{aligned}$$

This is because $(-1)^{x_i}$ get value 1 with probability $\frac{1}{2}$ and -1 otherwise. Thus $E[\chi_S] = 0$.

For (4), we note that

$$\begin{aligned} \chi_S(x) \chi_S(y) &= \prod_{i \in S} (-1)^{x_i + y_i} \\ &= \prod_{i \in S} (-1)^{(x_i + y_i) \bmod 2} \\ &= \chi_S(x + y). \end{aligned}$$

This completes the proof. ■

Let $\mathcal{F} = \{f \mid f : \{0, 1\}^n \rightarrow \mathfrak{R}\}$. Clearly, \mathcal{F} is a vector space of dimension 2^n over the field \mathfrak{R} . We will prove now a claim which demonstrates the importance of the set B .

Claim 5 B is a basis for \mathcal{F} .

Proof:

Clearly, $|B| = 2^n$ as the dimension of \mathcal{F} . Thus, it is sufficient to prove that the χ_S are independent. Assume that

$$\chi_T = \sum_{S \neq T} c_S \chi_S$$

then clearly, if we mulitple every side in this equation by χ_T , then the expectations under the uniform distribution of the expressions must be equal.

But from the last claim, we get:

$$\begin{aligned}
1 &= E[\chi_T \chi_T] \\
&= E\left[\sum_{S \neq T} c_S \chi_S \chi_T\right] \\
&= \sum_{S \neq T} E[c_S \chi_S \chi_T] \\
&= 0.
\end{aligned}$$

A contradiction. Thus, the set B is independent, and the claim follows. ■

Corollary 6 *Every function $f \in \mathcal{F}$ can be expressed as $\sum_S \widehat{f}_S \chi_S$. The $\{\widehat{f}_S\}$ are called the Fourier coefficients of f .*

We note that

$$f(x) \chi_{S_0}(x) = \widehat{f}_{S_0} + \sum_{S \neq S_0} \widehat{f}_S \chi_{S \Delta S_0}(x)$$

and thus

$$E[f(x) \chi_{S_0}(x)] = \widehat{f}_{S_0}.$$

One way to learn f in $\text{PAC}_U(\text{MQ})$ is to find estimations q_S of all \widehat{f}_S and output the hypothesis $h = \sum_S q_S \chi_S$. Two problems arise: The number of coefficients is 2^n , so we can't learn all the coefficients in polynomial time. In addition, The function h may not be boolean where f is a boolean function. But before, how can we estimate \widehat{f}_S ?

By the Hoeffding - Chernoff theorem we can approximate the value of one coefficient in the following way:

For a given λ and δ we denote $\max_x |f(x)| = B$ and we take

$$m = \frac{2B^2}{\lambda^2} \ln\left(\frac{2}{\delta}\right)$$

examples $(x_1, f(x_1)), \dots, (x_m, f(x_m))$. We denote

$$q = \frac{\sum_{i=1}^m f(x_i) \chi_S(x_i)}{m}$$

and then by Hoeffding - Chernoff theorem we get

$$\Pr [|q - E[f(x)\chi_{S_0}(x)]| > \lambda] \leq 2e^{-\frac{2\lambda^2 m}{4B^2}} = \delta$$

as we want.

As we have mentioned before we have an exponential number of coefficients, so we can't learn all the coefficients. Thus, we should choose only the heavy coefficients and learn only them.

We finish this section by a version of a well known theorem from algebra:

Theorem 7 (Parseval) *If f is a function in \mathcal{F} and $f = \sum_S \hat{f}_S \chi_S(x)$ then $E[f^2] = \sum_S \hat{f}_S^2$.*

Proof:

We compute the expectation, using claim 4.

$$\begin{aligned} E[f^2] &= E \left[\sum_S \hat{f}_S \chi_S \cdot \sum_T \hat{f}_T \chi_T \right] \\ &= E \left[\sum_{S,T} \hat{f}_S \hat{f}_T \chi_S \chi_T \right] \\ &= \sum_{S,T} E \left[\hat{f}_S \hat{f}_T \chi_S \chi_T \right] \\ &= \sum_{S,T} \hat{f}_S \hat{f}_T E[\chi_{S \Delta T}] \\ &= \sum_S E[\hat{f}_S^2] \\ &= \sum_S \hat{f}_S^2. \end{aligned}$$

The theorem now follows. ■

Corollary 8 *If f is a boolean function then $E[f^2] = \sum_S \hat{f}_S^2 = 1$.*

3.3 The learning algorithm

The idea of the algorithm is to learn only the heavy coefficients, those which satisfy $|\widehat{f}_S| \geq \theta$ for a given θ . The method uses the "Divide and Conquer" strategy. We want to find all the pathes to the heavy leaves in the tree in figure 1, where each leaf represent one coefficient, and every node has a weight that is the sum of the coefficients in the leaves of its subtree. As we will show, we can estimate this weight. The algorithm starts from the root and at each level proceeds to the next level to all children with weight $\geq \theta^2$. Note that if the weight of a node is less than θ^2 , then all the coefficients $|\widehat{f}_S|$ in this subtree are less than θ . As we will show, there are polynomial number of heavy nodes at each level, and thus the algorithm runs in polynomial time.

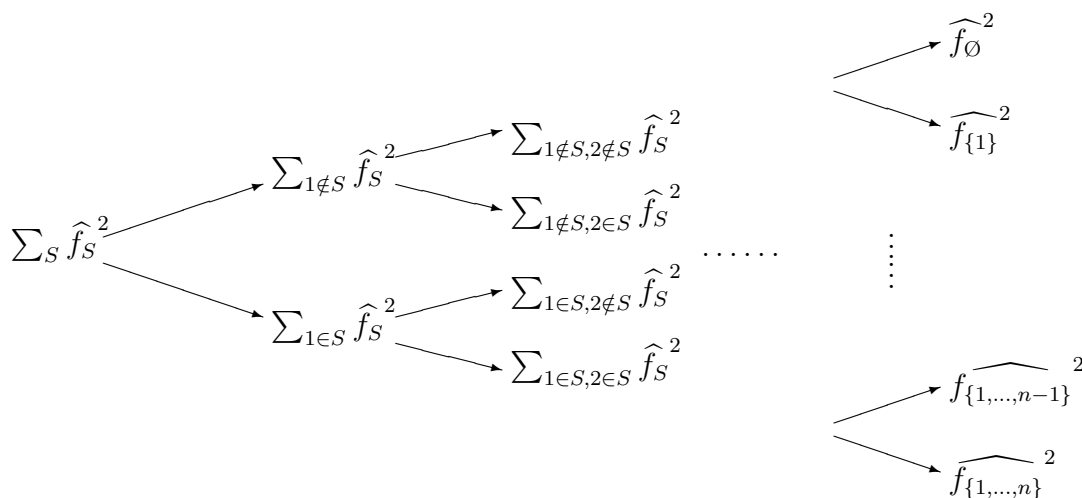


Figure 3.1: The tree of Fourier coefficients

We start by proving that the number of heavy leaves is polynomially bounded.

Claim 9 *Let f be a function such that $\max_x |f(x)| = B$, and let $\theta > 0$. Then at each level of the tree there are at most $\frac{B^2}{\theta^2}$ nodes that their weight are greater or equal to θ .*

Proof:

Clearly, $E[f^2] \leq B^2$, and then from Parsaval theorem, we get

$$\sum_S \widehat{f}_S^2 \leq B^2.$$

Suppose that at some level we have more than $\frac{B^2}{\theta^2}$ nodes with weight greater than θ at the same level. Then the sum of all the coefficients in the tree at this level is:

$$\sum_S \widehat{f}_S^2 > \theta^2 \cdot \frac{B^2}{\theta^2} = B^2.$$

A contradiction. The claim follows. ■

The second step is to show how to compute the weight of a node of the tree in figure 1. Take $m < n$ and fix $T \subseteq \{1, \dots, m\}$. We would like to compute the weight of the node which has all the coefficients \widehat{f}_S which have the following property: If $S \subseteq \{1, \dots, n\}$ is partitioned to (S_1, S_2) when $S_1 \subseteq \{1, \dots, m\}$ (m is a constant) and $S_2 \subseteq \{m+1, \dots, n\}$, then $S_1 = T$. In other words, we want to compute

$$\sum_{S_2 \subseteq \{m+1, \dots, n\}} \widehat{f_{T \cup S_2}}^2.$$

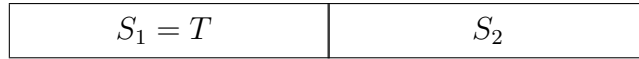


Figure 3.2: The partition of S to 2 parts

Let

$$x^m = (x_1, \dots, x_m)$$

and

$$x^{m+1, \dots, n} = (x_{m+1}, \dots, x_n).$$

We denote

$$x = x^m \cdot x^{m+1, \dots, n}.$$

We note that

$$\begin{aligned}
E_{x^m}[f(x^m x^{m+1\dots n})\chi_T(x^m)] &= E_{x^m}\left[\sum_S \widehat{f}_S \chi_S(x^m x^{m+1\dots n})\chi_T(x^m)\right] \\
&= E_{x^m}\left[\sum_S \widehat{f}_S \chi_{S_1}(x^m)\chi_{S_2}(x^{m+1\dots n})\chi_T(x^m)\right] \\
&= \sum_S \widehat{f}_S \chi_{S_2}(x^{m+1\dots n})E_{x^m}[\chi_{S\Delta T}(x^m)] \\
&= \sum_{S_2} \widehat{f_{T\cup S_2}} \chi_{S_2}(x^{m+1\dots n}).
\end{aligned}$$

Moreover, if we define

$$w = x^{m+1\dots n}, y = x^m$$

then we know from Parsaval theorem that:

$$\begin{aligned}
\sum_{S_2} \widehat{f_{T\cup S_2}}^2 &= E_w[E_y[f(wy)\chi_T(y)]^2] \\
&= E_w[E_y[f(wy)\chi_T(y)]E_z[f(zw)\chi_T(z)]] \\
&= E_{w,y,z}[f(yw)f(zw)\chi_T(y+z)].
\end{aligned}$$

The last expression can be computed by choosing uniformly w (of length $n - m + 1$) and y, z (of length m) and use the MQ oracle. We note that $|f(yw)f(zw)\chi_T(y+z)| \leq B^2$ and then we can approximate the value of expectation using Hoeffding - Chernoff theorem. Take

$$\lambda = \frac{\theta^2}{3}, m = \frac{9B^2}{\theta^2} \ln\left(\frac{2nB^2}{\delta\theta^2}\right).$$

Now, by Hoeffding - Chernoff theorem, we can bound the probability that we don't continue to search in a specific heavy subtree by:

$$\begin{aligned}
\Pr\left[\left|\frac{\sum_{i=1}^m X_i}{m} - p\right| > \lambda\right] &< 2e^{-\frac{9B^2\theta^2}{9B^2\theta^2} \ln\frac{2nB^2}{\delta\theta^2}} \\
&= \frac{\delta\theta^2}{nB^2}.
\end{aligned}$$

But according to Claim 9, there is at most $\frac{nB^2}{\theta^2}$ heavy nodes in the tree, and thus with probability at least $1 - \delta$ we find all the heavy coefficients of f .

Therefore, we can find all the heavy fourier coefficients in $\text{poly}(\theta, \log(\frac{1}{\delta}), B)$ time complexity.

We can use this method to present the algorithm:

The KM algorithm

1. Find all S such that $|\widehat{f}_S| \geq \theta$ by searching in the tree of the fourier coefficients. For each such S do $\Pi \leftarrow S$.
2. Compute q_s for every $s \in \Pi$ such that $|q_s - \widehat{f}_S| \leq \lambda$.
3. Define $g(x) \equiv \sum_{s \in \Pi} q_s \chi_S(x)$, and output $h(x) = \text{sign}(g(x))$.

Figure 3.3: Algorithm KM

Definition 2 For $f \in \mathcal{F}$ we define $L_1(f)$ to be $\sum_S |\widehat{f}_S|$.

Theorem 10 We can learn a boolean function in the $\text{PAC}_U(MQ)$ model in $\text{poly}(\log(\frac{1}{\delta}), \frac{1}{\epsilon}, L_1(f))$ time complexity.

Proof:

We use the algorithm that we described. Take

$$\theta = \frac{\epsilon}{2L_1(f)}$$

and

$$\lambda = \sqrt{\frac{\epsilon}{2}} \theta.$$

We denote Π to be the set of all the coefficients that the algorithm learns , and g to be the function from step 3 in the algorithm. From claim 9,

$$|\Pi| \leq \frac{1}{\theta^2}.$$

We note that

$$f - g = \sum_{S \notin \Pi} \widehat{f}_S \chi_S(x) + \sum_{S \in \Pi} (\widehat{f}_S - q_S) \chi_S(x).$$

Therefore, from Parsaval theorem we get:

$$\begin{aligned} E [(f - g)^2] &= \sum_{S \notin \Pi} \widehat{f}_S^2 + \sum_{S \in \Pi} (\widehat{f}_S - q_S)^2 \\ &\leq \theta \sum_S |\widehat{f}_S| + |\Pi| \lambda^2 \\ &\leq \theta L_1(f) + \frac{\lambda^2}{\theta^2} \\ &= \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon. \end{aligned}$$

If $f \neq h$ then

$$\frac{(f - h)^2}{4} = 1 \leq (f - g)^2$$

and if $f = h$ then

$$\frac{(f - h)^2}{4} = 0 \leq (f - g)^2.$$

Therefore

$$E \left[\frac{(f - h)^2}{4} \right] \leq E [(f - g)^2].$$

Then

$$\begin{aligned} \Pr [f \neq h] &= E \left[\frac{(f - h)^2}{4} \right] \\ &\leq E [(f - g)^2] \\ &\leq \epsilon. \end{aligned}$$

The theorem now follows. ■

We now show that decision trees have small L_1 and therefore are learnable with the KM algorithm.

Claim 11 *if $f, g \in \mathcal{F}$ and T is a term then*

1. $L_1(f + g) \leq L_1(f) + L_1(g)$
2. $L_1(T) = 1$

Proof:

1. We have

$$\begin{aligned}
 L_1(f + g) &= L_1\left(\sum_s (\widehat{f}_s + \widehat{g}_s) \chi_s\right) \\
 &= \sum_s |\widehat{f}_s + \widehat{g}_s| \\
 &\leq \sum_s |\widehat{f}_s| + \sum_s |\widehat{g}_s| \\
 &\leq L_1(f) + L_1(g).
 \end{aligned}$$

2. Assume that T is a term of size s . Without loss of generality, we can write

$$T = x_1 \dots x_s.$$

We note that every x_i :

$$x_i = \frac{1 - (-1)^{x_i}}{2}.$$

Therefore

$$\begin{aligned}
 T &= \prod_{A \subseteq \{1, \dots, s\}} \frac{1 - (-1)^{x_i}}{2} \\
 &= \sum_{A \subseteq \{1, \dots, s\}} \pm \frac{1}{2^s} \chi_A
 \end{aligned}$$

and thus,

$$L_1(T) = \frac{2^s}{2^s} = 1.$$

This completes the claim. ■

As we learned, the $\text{size}_{DT}(f)$ is the number of leaves in the DT representation of f .

Corollary 12 *The KM algorithm can learn a decision tree of size s in $\text{poly}(\log(\frac{1}{\delta}), \frac{1}{\epsilon}, s)$ time complexity.*

Proof:

As we learned before, each decision tree f can be represented as sum of terms:

$$f = T_1 + \dots + T_m.$$

Each term T_i correspond to some leaf labeled with 1. We change the functions' range to $\{-1, 1\}$ and get:

$$f' = 1 - 2f$$

and then from the last claim we get:

$$\begin{aligned} L_1(f') &\leq 1 + 2L_1(f) \\ &\leq 1 + 2(L_1(T_1) + \dots + L_1(T_m)) \\ &\leq 2m + 1. \end{aligned}$$

The proof now follows from theorem 10. ■

Remark: There is a DNF expression which its L_1 's size is exponential of n , so the algorithm can't help us to learn DNF. However, in the next section we will show how to use similiar techniques to weak learn DNF (we will define what a weak learning is in the next section). In the next lecture note we will see how we can use this weak learning to learn DNF in the $\text{PAC}_U(\text{MQ})$ model in polynomial time.

3.4 Weak learning of DNF

Definition 3 A function h is called a weak hypothesis for f if

$$\Pr[f \neq h] \leq \frac{1}{2} - \gamma$$

where $\gamma = \frac{1}{\text{poly}(n, 1/\epsilon)}$.

We first present a claim that shows a connection between $\Pr[f \neq h]$ and $E[fh]$.

Claim 13 $\Pr[f \neq h] = \frac{1}{2} - \frac{1}{2}E[fh]$.

Proof:

We note that

$$E[fh] = 1 \cdot \Pr[f = h] + (-1) \cdot \Pr[f \neq h].$$

In addition,

$$\Pr[f = h] = 1 - \Pr[f \neq h]$$

and thus we get

$$E[fh] = 1 - 2\Pr[f \neq h].$$

Therefore:

$$\Pr[f \neq h] = \frac{1}{2} - \frac{1}{2}E[fh].$$

The claim now follows. ■

Corollary 14 $\Pr[f \neq h] \leq \epsilon$ if and only if $E[fh] \geq 1 - 2\epsilon$

If we have a hypothesis h , we say that h is a weak hypothesis of f if

$$E[fh] > \frac{1}{\text{poly}(n, 1/\epsilon)}.$$

We say that h is a strong learning of f if $E[fh]$ is close to 1.

We present now three theorems. Then we will prove the third one, and in the next chapter it will be used in order to learn DNF in polynomial time.

Theorem 15 *DNF (of polynomial size) is a weak learnable in polynomial time under the uniform distribution.*

If for every f in a DNF form with $s = \text{poly}(n)$ we prove that there is a function h such that

$$E[fh] \geq \frac{1}{\text{poly}(n, s)}$$

then from claim 13, we get:

$$\Pr[f \neq h] \leq \frac{1}{2} - \frac{1}{\text{poly}(n, s)}$$

which is a weak learning. Thus it is sufficient to prove:

Theorem 16 *For every DNF with s terms there is S such that*

$$|\hat{f}_S| = |E_U[f\chi_S]| \geq \frac{1}{3s}.$$

Then χ_S or $-\chi_S$ is a weak learner, and we can find it with the KM algorithm.

Clearly, it is sufficient to prove the theorem for every distribution D :

Theorem 17 *For every DNF with s terms, and for every distribution D there is χ_S such that $|E_D[f\chi_S]| \geq \frac{1}{3s}$.*

Proof:

Let f be a function with the form $T_1 \vee T_2 \vee \dots \vee T_s$.

Then:

$$\begin{aligned} \Pr[f = 1] &= \Pr[T_1 = 1 \vee T_2 = 1 \vee \dots \vee T_s = 1] \\ &\leq \sum_{i=1}^s \Pr[T_i = 1]. \end{aligned}$$

Thus, there is i such that

$$\Pr[T_i = 1] \geq \frac{\Pr[f = 1]}{s}.$$

Without loss of generality T_i is montone. Thus:

$$\begin{aligned} \frac{\Pr [f = 1]}{s} &= \Pr_D [T_i = 1] \\ &= E_D [T_i] \\ &= E_D [T_i f]. \end{aligned}$$

As we have mentioned before,

$$x_i = \frac{1 - (-1)^{x_i}}{2}$$

and thus

$$\begin{aligned} T &= \prod_{A \subseteq \{1, \dots, s\}} \frac{1 - (-1)^{x_i}}{2} \\ &= E_{S \subseteq \{1, \dots, n\}} [(-1)^{|S|} \chi_S(x)]. \end{aligned}$$

Therefore,

$$\begin{aligned} E_D [T_i f] &= E_D [E_{S \subseteq \{1, \dots, n\}} [(-1)^{|S|} \chi_S(x)] f] \\ &= E_{S \subseteq \{1, \dots, k\}} [(-1)^{|S|} E_D [\chi_S(x) f(x)]] . \end{aligned}$$

Thus there is S such that

$$|E_D [\chi_S(x) f(x)]| \geq \frac{\Pr [f = 1]}{s}.$$

If $\Pr [f = 1] \leq \frac{1}{3}$ then $\Pr [f = 0] \geq \frac{2}{3}$ and as a result $-\chi_\phi$ is a weak learner of f . Otherwise $\Pr [f = 1] \geq \frac{1}{3}$ and thus

$$E_D [\chi_S(x) f(x)] \geq \frac{1}{3s}.$$

This completes the theorem. ■

References

[KM] Kushilevitz, E., and Mansour, Y., "Learning decision trees using the fourier spectrum", SIAM Journal on Computing, Vol. 22, No. 6, pp. 1331-1348, 1993