CSE 250: Machine Learning Theory Spring 2017

Part 4

Instructor: Raef Bassily Scribe: Andrew Leverentz

4.1 Weak versus Strong Learnability

Definition 4.1 (γ-weak learner)**.** Suppose γ be some small number bounded away from 1/2, say $\gamma \in (0, \frac{1}{4}]$. An algorithm A is a γ-weak learner for a hypothesis class $\mathcal{H}$ if there is a function $n_{\mathcal{H}} : (0,1) \to \mathbb{N}$ such that for every $0 < \delta < 1$ and every distribution $\mathcal{D}$, given $n \geq n_{\mathcal{H}}(\delta)$ i.i.d. examples from $\mathcal{D}$, w.p. $\geq 1 - \delta$, A outputs a hypothesis $h \in \mathcal{H}$ with

$$\text{err}(h; \mathcal{D}) \leq \frac{1}{2} - \gamma.$$

Informally, a weak learner merely has to be a little better than random guessing.

Definition 4.2 (Strong learner)**.** An algorithm A is a strong learner for a hypothesis class $\mathcal{H}$ if there is a function $n_{\mathcal{H}} : (0,1) \to \mathbb{N}$ s.t. for every $0 < \varepsilon, \delta < 1$, and for every distribution $\mathcal{D}$, given $n \geq n_{\mathcal{H}}(\varepsilon, \delta)$ i.i.d. examples from $\mathcal{D}$, w.p. $\geq 1 - \delta$, A outputs $h \in \mathcal{H}$ with

$$\text{err}(h; \mathcal{D}) \leq \varepsilon.$$

Question: Given a weak learner for a class $\mathcal{H}$, can we find an efficient method that produces a strong learner for a possibly more complex class $\widetilde{\mathcal{H}}$?

In the transformation from weak to strong learnability, we are concerned with boosting the accuracy, and we are allowed to output a more "complex" or "powerful" hypothesis which is normally constructed based on several hypotheses (or, "predictions") generated from the weak learner.

4.2 Boosting

Generally, the boosting paradigm adheres to the following outline:

- Round 1: $h_1 \leftarrow \text{WeakLearner}(\mathcal{D}_1)$ *(The weak learner is given i.i.d. examples from $\mathcal{D}_1$.)*
- Round 2: $h_2 \leftarrow \text{WeakLearner}(\mathcal{D}_2)$
- $\vdots$
- Round T: $h_T \leftarrow \text{WeakLearner}(\mathcal{D}_T)$
- Return: weighted majority of $h_1, \ldots, h_T$:

$$\tilde{h}(x) = \text{sign}(\alpha_1 h_1(x), \ldots, \alpha_T h_T(x)).$$

Here, $\mathcal{D}_1 = \mathcal{D}$ is the original distribution. For $t > 1$, $\mathcal{D}_t$ is a modification of $\mathcal{D}_{t-1}$, with higher weight on the examples that were misclassified by h_{t-1}.

The weights $\alpha_1, \ldots, \alpha_T$ are chosen such that more weight is given to "more accurate" hypotheses.

Note that $\tilde{h}$ does not necessarily belong to the "base" hypothesis class $\mathcal{H}$ associated with the weak learners. It is a linear classifier composed over T weak hypotheses.

Why would we want to use boosting?

Controlling the bias-complexity tradeoff Remember that our goal is to output a hypothesis h_S that has small true risk $\mathrm{err}(h_S; \mathcal{D})$. Note that

$$\mathrm{err}(h_S; \mathcal{D}) = \left(\mathrm{err}(h_S; \mathcal{D}) - \min_{h \in \mathcal{H}} \mathrm{err}(h; \mathcal{D}) \right) + \min_{h \in \mathcal{H}} \mathrm{err}(h; \mathcal{D}).$$

The first term (in parentheses) is the "estimation error" or "bias," and the second term is the "approximation error."

- Using a simpler hypothesis space typically increases the approximation error (more constrained search space) but decreases the estimation error (lower VC dimension).
- In contrast, using a richer hypothesis space typically decreases the approximation error (less constrained search space) but increases the estimation error (higher VC dimension).

This is the "bias-complexity tradeoff." Boosting lets us smoothly control this tradeoff by changing the value of T.

Better computational efficiency Boosting allows us to run a very simple (and therefore likely efficient) algorithm T times, where T is often small.

4.3 Adaboost ("Adaptive Boosting")

Adaboost is a particular boosting method due to Schapire and Freund (1995).

Let $\mathcal{Y} = \{-1, +1\}$ and let $S = \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathcal{X} \times \mathcal{Y}$. We will construct "artificial" distributions over S itself (as if S is the sample space). That is, we will treat S as a discrete distribution where probability mass only occurs at the observed sample points.

The distributions will be generated in an adaptive fashion. The initial distribution is a uniform distribution over S:

$$\begin{aligned} \mathcal{D}^{(1)} &= \left(\mathcal{D}_1^{(1)}, \mathcal{D}_2^{(1)}, \ldots, \mathcal{D}_n^{(1)} \right) \\ &= \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n} \right). \end{aligned}$$

The Adaboost algorithm is as follows:

function AdaBoost(WL, T)
  Inputs: Weak learner WL, number of rounds T.
  Initialize $\mathcal{D}^{(1)} = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right)$
  for $t = 1, \ldots, T$ **do**
    $h_t \leftarrow \mathrm{WL}(\mathcal{D}^{(t)}, S)$
    (That is, WL *gets points from S, sampled with repetition using weights specified by $\mathcal{D}^{(t)}$.)*
    $\varepsilon_t \leftarrow \sum_{i=1}^{n} \mathcal{D}_i^{(t)} \, 1(h_t(x_i) \neq y_i)$ *(Note: We expect $\varepsilon_t \leq \frac{1}{2} - \gamma$.)*
    $\alpha_t \leftarrow \frac{1}{2} \ln \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right)$
    Update the discrete distribution:

$$\mathcal{D}_i^{(t+1)} \leftarrow \frac{\mathcal{D}_i^{(t)} \exp(-\alpha_t \, y_i \, h_t(x_i))}{\sum_{j=1}^{n} \mathcal{D}_j^{(t)} \exp(-\alpha_t \, y_j \, h_t(x_j))}$$

end for
return $\tilde{h}$ defined as

$$\tilde{h}(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

end function

Remark 4.3. In the above algorithm, $\mathcal{D}_i^{(t+1)}$ is proportional to

$$\mathcal{D}_i^{(t)} \cdot \begin{cases} \sqrt{\dfrac{\varepsilon_t}{1-\varepsilon_t}}, & \text{if } x_i \text{ is classified correctly by } h_t, \\ \sqrt{\dfrac{1-\varepsilon_t}{\varepsilon_t}}, & \text{otherwise.} \end{cases}$$

This means that misclassified points tend to get weighted more heavily in the next round.

Also, the error of h_t w.r.t. $\mathcal{D}^{(t+1)}$ is exactly $\frac{1}{2}$:

$$\operatorname{err}(h_t; \mathcal{D}^{(t+1)}) = \frac{1}{2}.$$

This essentially forces Adaboost to output a new hypothesis in the next round since, as noted above, we expect $\varepsilon_t \leq \frac{1}{2} - \gamma$.

Example 4.4. Suppose $S \subset \mathbb{R}^2$ and $\mathcal{H}$ = horizontal or vertical half-planes (a.k.a. "decision stumps").

- Sample from $\mathcal{D}^{(1)}$ = uniform over S.
- Run weak learner on this sample to obtain h_1.
- Identify misclassified points
- Obtain $\mathcal{D}^{(t+1)}$ by assigning higher weight to these misclassified points.
- Next round: sample using $\mathcal{D}^{(t+1)}$ and run weak learner, obtaining h_{t+1}.

Theorem 4.5. Suppose $\varepsilon_t \leq \frac{1}{2} - \gamma$ for all $t \in [T]$ w.p. 1. Then, the training error of $\tilde{h}$ over the training set S is

$$\widehat{\operatorname{err}}(\tilde{h}; S) \leq \exp(-2\gamma^2 T).$$

Note: the premise above is stronger than what we have, but we will soon see how to generalize it to the case where the probability is $1 - \delta$.

Proof. For convenience, define

$$f(x) = \sum_{t=1}^{T-1} \alpha_t h_t(x), \text{ and}$$
$$z_t = \sum_{j=1}^{n} \mathcal{D}_j^{(t)} \exp(-\alpha_t \, y_j \, h_t(x_j)).$$

Thus, z_t is the normalizing factor in the expression for $\mathcal{D}^{(t)}$.

Then, note that we can expand the expression for the final distribution to obtain

$$\begin{aligned}\mathcal{D}_i^{(T)} &= \frac{1}{n} \cdot \frac{\exp\left(-y_i \sum_{t=1}^{T-1} \alpha_t \, h_t(x_i)\right)}{\prod_{t=1}^{T-1} z_t} \\ &= \frac{1}{n} \cdot \frac{\exp\left(-y_i \, f(x_i)\right)}{\prod_{t=1}^{T-1} z_t}\end{aligned}$$

Furthermore,

$$\begin{aligned}\widehat{\mathrm{err}}(\tilde{h}; S) &= \frac{1}{n} \sum_{i=1}^{n} 1(y_i f(x_i) \leq 0) \\ &\leq \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f(x_i)} && \text{(because } 1(u \geq 0) \leq e^u \text{ for any } u\text{)} \\ &= \sum_{i=1}^{n} \mathcal{D}_i^{(T)} \prod_{t=1}^{T-1} z_t \\ &= \prod_{t=1}^{T-1} z_t \sum_{i=1}^{n} \mathcal{D}_i^{(T)} \\ &= \prod_{t=1}^{T-1} z_t.\end{aligned}$$

Now, we just need to bound the product $\prod_{t=1}^{T-1} z_t$. We can rewrite each factor z_t as

$$\begin{aligned}z_t &= \sum_{i=1}^{n} \mathcal{D}_i^{(t)} \exp(-\alpha_t \, y_i \, h_t(x_i)) \\ &= \sum_{i \,:\, h_t \text{ correct on } (x_i, y_i)} \mathcal{D}_i^{(t)} \exp(-\alpha_t \, y_i \, h_t(x_i)) \\ &\quad + \sum_{i \,:\, h_t \text{ incorrect on } (x_i, y_i)} \mathcal{D}_i^{(t)} \exp(-\alpha_t \, y_i \, h_t(x_i)) \\ &= \sum_{i \,:\, h_t \text{ correct on } (x_i, y_i)} \mathcal{D}_i^{(t)} \exp(-\alpha_t) \\ &\quad + \sum_{i \,:\, h_t \text{ incorrect on } (x_i, y_i)} \mathcal{D}_i^{(t)} \exp(\alpha_t) \\ &= \sum_{i \,:\, h_t \text{ correct on } (x_i, y_i)} \mathcal{D}_i^{(t)} \sqrt{\frac{\varepsilon_t}{1 - \varepsilon_t}} \\ &\quad + \sum_{i \,:\, h_t \text{ incorrect on } (x_i, y_i)} \mathcal{D}_i^{(t)} \sqrt{\frac{1 - \varepsilon_t}{\varepsilon_t}}.\end{aligned}$$

Then, we can rewrite this in terms of probabilities:

$$\begin{aligned}
z_t &= \sqrt{\frac{\varepsilon_t}{1-\varepsilon_t}} P_{(x,y)\sim\mathcal{D}^{(t)}}(h_t(x) = y) + \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}} P_{(x,y)\sim\mathcal{D}^{(t)}}(h_t(x) \neq y) \\
&= \sqrt{\frac{\varepsilon_t}{1-\varepsilon_t}}(1-\varepsilon_t) + \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}\varepsilon_t \\
&= 2\sqrt{\varepsilon_t(1-\varepsilon_t)} \\
&\leq 2\sqrt{(1/2-\gamma)(1/2+\gamma)} \\
&= 2\sqrt{1/4-\gamma^2} \\
&\leq e^{-2\gamma^2}.
\end{aligned}$$

Hence,

$$\widehat{\mathrm{err}}(\tilde{h}; S) \leq \prod_{t=1}^{T-1} z_t \leq e^{-2\gamma^2 T}.$$

□

If we generalize the above theorem (that is, if we merely assume ε_t is small w.p. $\geq 1-\delta$), the conclusions now occur w.p. $1 - T\delta$, since δ represents the probability of failure in one round, and we can apply a union bound over the full T-step procedure.

Corollary 4.6. If WL consistently (w.p. 1) returns hypotheses with error $\leq \frac{1}{2} - \gamma$, then the final hypothesis $\tilde{h}$ after $T = \frac{1}{2\gamma^2} \ln \frac{1}{\varepsilon}$ rounds is guaranteed to have $\widehat{\mathrm{err}}(\tilde{h}; S) \leq \varepsilon$.

Corollary 4.7. If WL returns (w.p. $\geq 1-\delta$) hypotheses with error $\leq \frac{1}{2} - \gamma$, then the final hypothesis $\tilde{h}$ after $T = \frac{1}{2\gamma^2} \ln \frac{1}{\varepsilon}$ rounds has $\widehat{\mathrm{err}}(\tilde{h}; S) \leq \varepsilon$ w.p. $\geq 1 - \frac{\delta}{2\gamma^2} \ln \frac{1}{\varepsilon}$.

Next, we want to bound (with high probability) the generalization error of $\tilde{h}$; that is, to bound

$$\widehat{\mathrm{err}}(\tilde{h}; S) - \mathrm{err}(\tilde{h}; \mathcal{D}),$$

where S is a set of n i.i.d. examples from the unknown distibution $\mathcal{D}$.

From the triangle inequality, note that

$$\mathrm{err}(\tilde{h}; \mathcal{D}) \leq \widehat{\mathrm{err}}(\tilde{h}; S) + \left| \widehat{\mathrm{err}}(\tilde{h}; S) - \mathrm{err}(\tilde{h}; \mathcal{D}) \right|.$$

If $\tilde{h}$ comes from a class $\tilde{\mathcal{H}}$ with finite VC dimension $\tilde{k}$, then

$$\begin{aligned}
P\left(\left| \widehat{\mathrm{err}}(\tilde{h}; S) - \mathrm{err}(\tilde{h}; \mathcal{D}) \right| > \varepsilon \right) &\leq P(\exists\, h \in \tilde{\mathcal{H}} : |\widehat{\mathrm{err}}(h; S) - \mathrm{err}(h; \mathcal{D})| > \varepsilon) \\
&\leq 4\, \hat{C}_{\tilde{\mathcal{H}}}(2n) e^{-\varepsilon^2 n/8} \\
&\leq 4 \left(\frac{en}{\tilde{k}} \right)^{\tilde{k}} e^{-\varepsilon^2 n/8}.
\end{aligned}$$

Theorem 4.8. Let $\mathcal{H}$ be a base hypothesis class (that is, the class associated with a weak learner) where $\mathrm{VC}(\mathcal{H}) = k$. The class

$$\tilde{\mathcal{H}} = \left\{ \tilde{h} : \forall x \in \mathcal{X}, \tilde{h}(x) = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t\, h_t(x) \right) \text{ and } \forall t \in [T], \alpha_t \in \mathbb{R}, h_t \in \mathcal{H} \right\}$$

has VC dimension

$$\mathrm{VC}(\tilde{\mathcal{H}}) \leq O(kT \log(kT)).$$

Remark 4.9. In boosting, the "input" hypothesis all come from $\mathcal{H}$, and we feed them into a hypothesis from class $\mathcal{G}$ = homogeneous linear classifiers. We can think of the class $\tilde{\mathcal{H}}$ as the composition of $\mathcal{G}$ applied to T instances of $\mathcal{H}$.

More generally, we could describe a compound structure using a directed acyclic structure of compositions of hypothesis classes.

Theorem 4.10. For any directed acyclic graph with T nodes where each node represents a hypothesis class $\mathcal{H}_i$ with $\text{VC}(\mathcal{H}_i) = k_i$, the equivalent compound hypothesis class $\tilde{\mathcal{H}}$ represented by the graph has VC dimension

$$\text{VC}(\tilde{\mathcal{H}}) \leq 2\left(\sum_{i=1}^{T} k_i\right) \log\left(e\sum_{i=1}^{T} k_i\right).$$

Proof. We will prove this by induction. Suppose the input is a set U of n unlabeled points: $U = \{x_1, \ldots, x_n\}$. Consider the composition of a hypothesis from H_3 applied to the output of two hypotheses, one from $\mathcal{H}_1$ and the other from $\mathcal{H}_2$.

The class $\mathcal{H}_1$ can generate at most $\hat{C}_{\mathcal{H}_1}(n)$ labelings on U. Similarly, the class $\mathcal{H}_2$ can generate at most $\hat{C}_{\mathcal{H}_2}(n)$ labelings on U.

Then, $\mathcal{H}_3$ has at most $\hat{C}_{\mathcal{H}_1}(n) \cdot \hat{C}_{\mathcal{H}_2}(n)$ inputs.

Fix $h_1 \in \mathcal{H}_1$ and $h_2 \in \mathcal{H}_2$; that is, fix a labeling $(h_1(x_1), \ldots, h_1(x_n))$ generated by $\mathcal{H}_1$ and a labeling $(h_2(x_1), \ldots, h_2(x_n))$ generated by $\mathcal{H}_2$.

The inputs to $\mathcal{H}_3$ can be written as:

- Input 1: $(h_1(x_1), h_2(x_1))$
- Input 2: $(h_1(x_2), h_2(x_2))$
- $\cdots$
- Input n: $(h_1(x_n), h_2(x_n))$

This is a set of "new" domain points of size n. Hence, $\mathcal{H}_3$ can generate at most $\hat{C}_{\mathcal{H}_3}(n)$ labelings on these particular inputs.

In total, the graph through $\mathcal{H}_3$ can generate at most $\hat{C}_{\mathcal{H}_1}(n) \cdot \hat{C}_{\mathcal{H}_2}(n) \cdot \hat{C}_{\mathcal{H}_3}(n)$ labels on U.

Proceeding inductively, we can show that the total number of labelings that the equivalent compound hypothesis class $\widetilde{\mathcal{H}}$ can generate is

$$\hat{C}_{\widetilde{\mathcal{H}}}(n) \leq \prod_{i=1}^{T} \hat{C}_{\mathcal{H}_i}(n).$$

To obtain our VC result, suppose that the largest set that can be shattered by $\widetilde{\mathcal{H}}$ has size k. The number of labelings when shattered is 2^k, and

$$\begin{aligned} 2^k &= \hat{C}_{\widetilde{\mathcal{H}}}(k) \\ &\leq \prod_{i=1}^{T} \hat{C}_{\mathcal{H}_i}(k) \\ &\leq \prod_{i=1}^{T} \left(\frac{ek}{k_i}\right)^{k_i}. \qquad \text{(by Sauer's lemma)} \end{aligned}$$

Taking the log of both sides,

$$k \leq \sum_{i=1}^{T} k_i \log_2 \left(\frac{ek}{k_i} \right).$$

Now, define

$$f(k) = k - \sum_{i=1}^{T} k_i \log_2 \left(\frac{ek}{k_i} \right) \leq 0.$$

Observe that $f(k)$ is an increasing function in k when $k \geq 2\sum_{i=1}^{T} k_i$. Suppose we set

$$k = k^* = 2 \left(\sum_{i=1}^{T} k_i \right) \log_2 \left(e \sum_{i=1}^{T} k_i \right).$$

We can show that $f(k^*) > 0$, violating an earlier inequality. Since f is increasing, all values of $k > k^*$ will also violate this inequality. Hence, $k \leq k^*$, which means

$$\mathrm{VC}(\widetilde{\mathcal{H}}) \leq 2 \left(\sum_{i=1}^{T} k_i \right) \log_2 \left(e \sum_{i=1}^{T} k_i \right).$$

□

Remark 4.11 (Boosting case)**.** For boosting, there are T base classes (all equal to $\mathcal{H}$), and one class $\mathcal{G}$ of *homogeneous* linear separators (with $\mathrm{VC}(\mathcal{G}) = T$).

The sum of all VC dimensions in the graph is $kT + T$. Applying the theorem to the compound class of boosted hypotheses,

$$\begin{aligned} \mathrm{VC}(\widetilde{\mathcal{H}}) &\leq 2(kT + T) \log_2 \left(e(kT + T) \right) \\ &= O(kT \log(kT)). \end{aligned}$$

4.4 Margin-Based Explanation of Boosting

In many real scenarios, Adaboost has shown an interesting behavior. In such scenarios, when we plot both the training error and test error versus the number of boosting rounds, T, we notice that the training error goes to zero and stays at zero as we increase T, and the test error (which is an approximate estimate for the true error based on a separate testing set) remains decreasing as we increase T even after the training error goes to zero before it eventually starts to rise. So, why is the test error keep decreasing with T (at least for a while)? Isn't this counter to the insight we get from the upper bound on the VC-dimension we derived above?

An elegant explanation for this behavior is due to [Schapire, Freund, Bartlett, Lee '97]. The explanation is shows the dependence of Adaboost on a quantity called *the margin*.
Define the margin on a data point (x, y) as

$$\mu(x, y) = y \sum_{t=1}^{T} \tilde{\alpha}_t h_t(x)$$

where for all $t \in [T]$, $\tilde{\alpha}_t$ is a normalized version of α_t in the Adaboost algorithm. That is,

$$\tilde{\alpha}_t = \frac{\alpha_t}{\sum_{k=1}^{T} \alpha_k}.$$

Note that, previously, what we cared about is the sign of $\mu(x_i, y_i)$ over the data points $(x_1, y_1), \ldots, (x_n, y_n)$. In particular, when $\mu(x, y) > 0$, this means that the final classifier (the weighted majority vote) correctly classifies x. It turns out that Adaboost behavior is tied to both the magnitude and the sign of $\mu(x_i, y_i)$, and not just the sign.

The magnitude of this quantity signifies how confident is the final classifier about the label it produces. This intuitively means that the larger this quantity is (given that it's positive), the better. But, how to nail down exactly the relatioship between Adaboost's performance and μ.

It was shown (e.g., the reference above) that Adaboost, in essence, is an iterative minimization algorithm, and the cost function it is trying to minimize is $\frac{1}{n}\sum_{i\in[n]} e^{-\mu(x_i,y_i)}$ over the choice of $h_1, \ldots, h_T$. (Note that, by defintion, $\mu(\cdot, \cdot)$ depends on $h_1, \ldots, h_T$).

So, even after all points $(x_1, y_1), \ldots, (x_n, y_n)$ are correctly classified and the training error becomes zero, (i.e., $\mu(x_i, y_i) > 0 \ \forall i \in [n]$), if we increase T beyond that point then Adaboost would try to derive the above cost function down by increasing the magnitude of the margin $\mu(x_i, y_i)$ for all points $(x_i, y_i), i \in [n]$. But, what implication does this have on the generalization error.

The following result characterizes the relationship between the margin and the generalization error. It conforms with our intuition: larger margin $\Rightarrow$ better generalization error.

Theorem 4.12. Let $\mathcal{D}$ be any distribution on $\mathcal{X} \times \{-1, +1\}$. Let $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a training set of n i.i.d. examples from $\mathcal{D}$. Let $\mathcal{H}$ denote the base hypothesis class where $VC(\mathcal{H}) = K$. Define

$$\tilde{\mathcal{H}} = \left\{ \tilde{h} : \forall x \in \mathcal{X}, \tilde{h}(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t \, h_t(x)\right) \text{ and } \forall t \in [T], \alpha_t \geq 0 \text{ s.t. } \sum_{t\in[T]} \alpha_t = 1, h_t \in \mathcal{H} \right\}$$

Let $0 < \delta < 1$. Then, with probability at least $1 - \delta$ over the choice of S, for all $\theta > 0$ and all $\tilde{h} \in \tilde{\mathcal{H}}$, we have

$$\text{err}(\tilde{h}; \mathcal{D}) \leq \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\left(\mu(x_i, y_i) < \theta\right) + O\left(\sqrt{\frac{K \log^2(n/K) + \theta^2 \log(1/\delta)}{\theta^2 m}}\right).$$