(4)

0,5

3° We could use a feature map or a kernel.

0,5

4° (a) $\nabla_w J = \nabla_w \frac{1}{4} \sum_n^N (\vec{w}.\vec{x} - y_n)^4 + \frac{1}{4} \lambda \|\vec{w}\|_2^4$

$= \sum_n^N (\vec{w}.\vec{x} - y_n)^3 \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{pmatrix} + \lambda \|\vec{w}\|^3$

0,25

(b) For the fit function we will change the w = ...
with this new formula.
Predict would not change.

0,75

(c) I would think that LinReg² would perform better because it is updated with the same and evaluated with the same idea.

5° (a) Instead of doing an epoch on N points we would only do it on [1, ..., batch-size].
fit (...):
for epoch in [1, ..., Nbre-epoch] do
for n in [1, ..., batch-Size] do.
w t = ...
Thus $\vec{\nabla}_\theta J = \vec{\nabla}_\theta \frac{1}{i} \sum_n^{batch\,size} (...)$

1

(b) Using mini-batch GD would help us avoid overfitting on outlier data because it would have fewer exemple of them.

0,5

(c) The fit time will be smaller.

0x

(1)

université PARIS-SACLAY

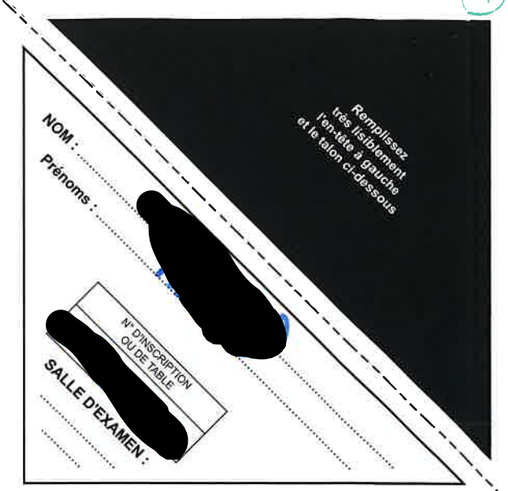
000060

ANNÉE : 2020

UNITÉ D'ENSEIGNEMENT : TC0

ÉPREUVE DE : Examen

NOTE	/20	Coefficient	Note affectée du coefficient
	19,5		



Si votre composition comporte plusieurs feuilles, numérotez-les 1/...

(6)

Analysis of experimental results :

1° a - The left one is the richer cause when increase λ the train score decreases.
The train score decreases because the regularization becomes stronger and it can not learn by heart the train set anymore.
The valid score increases because the model generalize better when not learning by heart.

1,5

b - The second one because we have the same result in validation when the model is less complexed.
Thus we gain computation time.

2° The left one seems to be the one complexed and with regularization because it looks like we have small pikes but small due to the regularization.
In the right one, we have a nice curve without any pikes.

0,5

PARTIE À RABATTRE

3°a - The train set and valid set have the same score, but it does not look like we have overfitting because the two are close and the validation does not decrease.

b - Maybe because weird data points that can not fit even with complexed model. For the validation it's the same idea.

c - We could expect a test score of <0.05.

4° (a) As being the separator between the two groups.

(b) As being the margin which describes which points are being the support ~~one~~ vectors.

(c) Case one to 4 do not seem to overfit but of course 1 and 2 are underfitting so they are not ok. But 1 seems to be a bit too simple. So 2 to 4 are ok and 1 is too simple and for 5 and 6 we overfit.

(d) I would pick 2 because even if we have some errors, I would expect that the red points could exist to the bottom right and it's the one that expect it the most.

5° He seems to learn on the test set about which model and hyperparameters are the best, whereas we should only do this on the validation set.

6° We should try to gather data that is not biased or we could increase the penality when the model becomes racist / sexist.

Yes !

Linear Regression

1° Linear regression is used when we want to predict scalar result using our features.

$J = \frac{1}{2} \sum_n^{\cdot} (f(x^{(n)}) - y)^2$ with $f(x^{(n)}) = \vec{w} \cdot \vec{x}^{(n)}$

We are doing a gradient descent as optimization algorithm.

```
def fit (X, y, η, w_0, maxEpoch):
    w = w_0
    for epoch in [1, ... maxEpoch] do
        w = w + η x X[n]
        for n in [1, ... size(X)] do
            w = w + η x (X[n] . w - y[n]) x X[n]
    return w

def predict (x, w):
    return x . w
```

2° $w_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ $\quad X = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 \end{pmatrix}$ with $x[0] = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

$w_1 = w_0 + \eta \times (\vec{x}[0] \cdot \vec{w} - y) \vec{x}[0]$

$= \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$w_2 = w_1 - 1 \times \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} - 2 \right) \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$

$w_3 = w_2 + \left(\begin{pmatrix} 1 \\ 2 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 2 \end{pmatrix} - 2 \right) \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} -3 \\ -6 \end{pmatrix}$

$w_4 = w_3 + \left(\begin{pmatrix} 1 \\ 2 \end{pmatrix} \cdot \begin{pmatrix} -3 \\ -6 \end{pmatrix} - 3 \right) \begin{pmatrix} -3 \\ -6 \end{pmatrix} = \begin{pmatrix} -56 \\ -102 \end{pmatrix}$

0,25 5° If we have a plane $\vec{w_{kk'}}$ when we visit k_k'', we are going to have an inner product being equal to 0.

6° For $\vec{x}^{test}$ we are going to have (1/0) or (0/1) with (1/0) saying that X is more likely to be k and (0/1) saying that X is more likely to be k'.
Thus we have $D \times \sum_{k=1}^{K} k_k$ predictions in total.

0,75 Predict will look like
predict = $\text{mode}\left(\sum_{k_k k'k''} y^{Pred}_{k''k_k''}\right)$ or

7°

8° As hyperparameter we have:
✓ - the definition of 6
✗ - the shape of our data set.
✓ - how many classes k do we have

0,5

université PARIS-SACLAY

000060

ANNÉE : 2020

UNITÉ D'ENSEIGNEMENT : TCO

ÉPREUVE DE : Examen

NOTE /20	Coefficient	Note affectée du coefficient

Remplissez très lisiblement l'en-tête à gauche et le talon ci-dessous

NOM :

Prénoms :

N° D'INSCRIPTION OU DE TABLE

SALLE D'EXAMEN :

Si votre composition comporte plusieurs feuilles, numérotez-les 2/2

4 Identifying the task

1° (a) The goal is to classify mushrooms as being poisonous or not.
(b) We should do supervised classification
(c) Our Label is the feature "edible / poisonous" and our features would be encoded as "one-hot".
So for each features such as "cap-shape" we would create n features where n is the number of possibilities.
(d) A decision Tree Classifier could work.
(e) We should make a metric where if a poisonous mushrooms is being predict as edible, it should punished 10 times more than the inverse.

1,25

2° (a) The goal is which team will win.
(b) We should do supervised classificaten
(c) Our Label will be the winner.
Our features are for the most numerical but for the encoding of the categorical one (= "Name" and "type") we will do hot encoding (except if the Name creates too many features).

0,75

PARTIE À RABATTRE

(d) A perceptron could do the work.
(e) Maybe if I have the too many we could have one issue.
And depending on which team attack Si

3° (a) Our goal is to predict temperatures
(b) We would have supervised regression.
(c) We could not encode it
As For the temperature, longitude and latitude they are already numerical so we have nothing to do, but for the time we could encode them as being the hour since the begining of 1900. ?
(d) We could use linear regression
(e) We could have a feature where we take the date (hour/month) and (the one not being transformed yet) and use the day and month to add a feature of the season.

1,25

4° (a) Our goal is to classify players into n roles.
(b) We would have supervised classification.
(c) We could remove the name because it has nothing to do with the position and encode with week foot and prefered foot as ... foot right or not (hot encoding). For the rest as it's numerical we have nothing to do.
(d) A Multi-Class perception could work.
(e) We could remove create labels being the combination of two positions which are similar.

0,75

No

3

3 OVO Classification

1°(a) If n is of class 1 then we have
$\sigma(\vec{w}_{12} \cdot \vec{x}_n) = (1,0)^T$
If n is of class 2 then we have
$\sigma(\vec{w}_{12} \cdot \vec{x}_n) = (0,-1)^T$

Thus we could have $t_{n,12}$ as being either $(1,0)^T$ or $(0,-1)^T$ but not $(0,0)^T$ nor $(1,-1)^T$.

X

No

(b) $\vec{w}_{21} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \vec{w}_{12}$ and $t_{n,21} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} t_{n,12}$

(c) $\sigma(\vec{w} \cdot \vec{x}_n) = \sum_k^K t_{nk}$
$t_{nRR'} = \begin{cases} (1,0)^T \text{ if } R \text{ is more Pitbely} \\ (0,1)^T \text{ if } R' \text{ is more Pitbely} \end{cases}$

2° We have $D \times K$ parameters.

3° $J = \frac{1}{N} \sum_n^N \left(\sum_d^D \left(x_n \cdot \vec{w}_d - y_n \right) \right)$

4°

0,5

2° We have $D \times \left(\sum_{k=1}^{K} k \right)$ parameters, because we can retrieve w_{21} from w_{12}.

0,25

3° $J = \frac{1}{N} \sum_n^N \left(\sum_{R=1}^{K} \left(\sum_{R'=R+1}^{K} \left(\sigma(x_n \cdot w_{RR'}) - t_{n,RR'} \right)^2 \right) \right)$

yes

0k.

4(a) $\nabla_\theta J(\theta, x) = \nabla_\theta \frac{1}{N} \sum_n^N \left(\sum_{R=1}^{K} \left(\sum_{R'=R+1}^{K} \left(\tanh(x_n \cdot w_{RR'}) - t_{n,RR'} \right)^2 \right) \right)$

$= \frac{1}{N} \sum_{n}^{N} \sum_{R=1}^{K} \sum_{R'=R+1}^{K} \left(\text{not } 2 \left(\tanh(x_n \cdot w_{RR'}) - t_{n,RR'} \right) \right)$

$\frac{\left(4(x_n) \right)}{\left(\cosh(x_n \cdot w_{RR'}) \right)^2}$

0,75

idea = coherent

(b) $\frac{8}{N} \tanh\left(\underset{1, \frac{K}{2}RR' \times D}{W^T} \underset{D,N}{X} \right) - \left(\underset{n, \frac{K}{2}RR'}{T} \right)^T$