7,75 (4)

Assuming we have: X-exp, a matrix $(N, D+1)$, where every row is the vector starting with a 1 and then the D features

2

predict:

def predict (X-ext, theta):

~~y_pred = θ * X~~

return ~~X ext~~ np.matmul (X.T, theta)

matrix mult. $X^T \cdot \theta$ ($N, D+1$; $D+1, 1$) → $N, 1$

$N, 1$

2. update $X = (1, 1, 2, 2)$ $w_0 = (1, 1)^T$. $\eta = 1$

$y = (1, 2, 2, 3)$ $N = 4$

we extend X: $X' = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 \end{bmatrix}$

$f(x) = \frac{1}{N} \sum_{i=1}^{N} \vec{\theta} \cdot x + b$ → $f(\vec{x}) = \vec{w} \cdot \vec{x}'$ = scalar de

$J(w, x) = \frac{1}{N} \sum_{i=1}^{n} (\vec{w} \cdot \vec{x}_i' - y_i)^2 = w^t x$

1,3

$\nabla_w J(w, x) = 2 \cdot \frac{1}{N} \sum_{i=1}^{n} (\vec{w} \cdot \vec{x}_i - y_i) \cdot \vec{x}_i$

UPDATE RULE: $w_{t+1} = w_t - \eta \cdot \nabla J(w, x) = w_t - \frac{2}{N} \cdot \sum_{i=1}^{n} (\vec{w} \cdot \vec{x}_i - y_i) \cdot \vec{x}_i$ ($\frac{2}{N} = 1/2$)

(1)

$i=1$ $f(x_1) = [1\ 1] \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 2 \rightarrow f(x_1) - y_1 = 1$

$i=2$ $f(x_2) = [1\ 1] \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 2 \rightarrow f(x_2) - y_2 = 0$

$i=3$ $f(x_3) = [1\ 1] \begin{bmatrix} 1 \\ 2 \end{bmatrix} = 3 \rightarrow f(x_3) - y_3 = 1$

$i=4$ $f(x_4) = [1\ 1] \begin{bmatrix} 1 \\ 2 \end{bmatrix} = 3 \rightarrow f(x_4) - y_4 = 0$

$w_1 = w_0 - \frac{1}{2}$ ~~...~~ $\left[1 \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} + 0 \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} + 1 \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} + 0 \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right] =$

$= w_0 - \frac{1}{2} \cdot \left[\begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right] =$

$= w_0 - \frac{1}{2} \begin{bmatrix} 2 \\ 3 \end{bmatrix} =$

$= \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 3/2 \end{bmatrix} = \begin{bmatrix} 0 \\ -1/2 \end{bmatrix}$

$w_1 = \begin{bmatrix} 0 \\ -1/2 \end{bmatrix}$

0,5

3. relation not linear. We could apply a feature map to the input, increase the dimensionality of the data.

$(w^T w)(w^T \cdot w)$

4. a) $J(w, \vec{x}_{n,n}) = \frac{1}{4} \sum_{n}^{N} (\vec{w} \cdot \vec{x}_n - y_n)^4 + \frac{1}{4} \cdot \lambda \|w\|^4$. Gradient descent. Lin g^2.

$\nabla_w J = 4 \cdot \frac{1}{4} \sum_{n}^{N} (\vec{w} \cdot \vec{x}_n - y_n)^3 \vec{x}_n + 4 \cdot \frac{1}{4} \cdot \lambda \cdot w \|w\|^2$

1

$\nabla_w J = (w^T X - y)^3 \cdot X + \lambda \cdot w \|w\|^2$

$w_{t+1} = w_t - \eta \cdot \Sigma (\vec{w} \cdot \vec{x} - y_n)^3 \cdot \vec{x}_n + \lambda \cdot w \|w\|^2$

$= w_t - \eta \cdot (w^T X - y)^3 \cdot X + \lambda \cdot w \cdot \|w\|^2$

b) There are no changes in the predict function, the function to predict is the same one, we just change the ~~...~~ cost function. We will change the fit function, we will recalculate J as the expression above and the update of theta, according to above ...

0,25

Notez avec exactitude votre numéro de table ou d'inscription.
Il est interdit au candidat de signer sa copie ou d'y mettre un signe quelconque pouvant identifier l'auteur de la copie.

université PARIS-SACLAY

000066

ANNÉE : ....

UNITÉ D'ENSEIGNEMENT : ....

ÉPREUVE DE : ....

NOTE /20	Coefficient	Note affectée du coefficient
18,25		

NOM : [redacted]

Prénoms : [redacted]

N° D'INSCRIPTION OU DE TABLE

SALLE D'EXAMEN :

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

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

(1)

(4)

① ANALYSIS OF EXPERIMENTAL RESULTS

1. regularization

a) To assess expressivity, we should take a look at the accuracy, in this case, the validation score. For the correct regularization, both models have similar expressivity, but in general, the right one has bigger expressivity, for smaller λ has better accuracy. The variations in a) are explained thanks to regularization. For small λ, there's not much regularization, our model is probably too complex for the data and clearly is overfitting, that's why we have such a big gap between train and validation scores. However, for big λ, when regularization is applied, we reduce the complexity of our model, we reduce overfitting and improve generalization, validation score and the gap between train and validation

0,5

b) If we take into account only train data, we would say the model on the left fits it better (is overfitting and has better train score). However, taking both training and validation sets into account, will move us conclude that, ~~however~~ with regularization, there's still difference between train and validation score, so we might still be overfitting our data (even for λ, we can see a small improvement ~~with~~ in validation with big regularization). Consequently, we are probably fitting quit good the data and not generalizing that well.

X

pick clearly!

So then?

Notez avec exactitude votre numéro de table ou d'inscription.
Il est interdit au candidat de signer sa copie ou d'y mettre un signe quelconque pouvant identifier l'auteur de la copie.

2. Figure 2. In this case a complex model might probably be a high-degree polynomial while a simpler one might have lower degree. In case c the line has more noticeable variations, different slopes, that makes us think it was generated with a more complex function (even if regularization makes it smoother, to not overfit). Whereas in d the function is more smooth, simply consisting of a low-degree polynomial.

So, c: complex with regularization / d: less complex no regularization.

3. Figure 3. In these case, for small model complexity, our validation error is bigger than the train, but as complexity increases the error decreases. This means that, at first, our model wasn't expressive enough, we were underfitting the data. When our model gets complex enough both errors are similar, there is No underfitting because both errors remain similar. b) The train error is not 0 because we are not perfectly fitting the data, otherwise we would have overfitting. Validation = 0 would mean a perfect model that can generalize perfectly, not very likely.

4. Figure 4:

a) The solid line is the border, the line that separates both sides (or all sides) of the plane. b) The dashed lines indicates where the margin starts (same case from the line to the border), indicating that all the points inside will generate support vectors.

c) Numerating from 1 to 6, from left-right and top-bottom $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$, 5 and 6 are clear cases of overfitting the borders are very complex and we even have more than one. We will probably have good performance in train but it won't generalize well. Number 4 and 3 have really big margins, what might be a sign for underfitting as well. Consequently, 2 looks a good model simple margin and probably good generalisation. Model 1 will also generalize well, even if the performance won't be as good as in 2.

d) I will take the risk of model 2 probably, the size of margin and the amount of points inside seems reasonable for both classes.

3. c) Since train error and validation error are similar (for complex enough models), we can expect a test error similar to both other values, there's no overfitting. However, for poor complex, test error will be similar to validation error.

5. Benji: what he calls test set might start becoming a second train set, since he is using it repeatedly to optimize hyper-parameters. He could do something better, like a cross-validation, even with this test set, but just to optimize the hyper parameters, making sure he's not overfitting, that there's no data leakage in his model.

6. Recruiting people: We could add some prior of our model tells us. We could try to use the Maximum a posteriori, add a priory probability estimation for the parameters (about the sex in races/sexs).

2 Linear Regression

1. Linear regression are models to perform a regression task, we want to predict continuous labels. In a regression task we have: • Input features ($\vec{x}$, N samples, in mat.) $X = \{\vec{x}_1, \dots, \vec{x}_N\}$ • Their corresponding output, or ground truth (y) • Model, with some parameters $\vec{\theta}(\theta_1 \dots \theta_D, b)$ that tries to predict the label $\hat{y}$ for each data point $\vec{x}$ inputed.

• To do the prediction we use a function $f_\theta(\vec{x})$ that can be either linear ($f(\vec{x}) = \vec{\theta} \cdot \vec{x} + b$) or polynomial ($f(\vec{x}) = \sum \theta_i x^i$). In order to have a good prediction, our model needs to define its parameters, so the predicted labels are as similar as possible to ground truths. To control/measure the similarity we have a cost function, least squares for example (LSE: $J(\theta, x) = \frac{1}{N}\sum_{i=1}^{N} (f(\vec{x}) - y)^2$). Ideally, we need to minimize this cost function and to do so we want our parameters $\vec{\theta}$ to be: $\theta^* = \arg\min_\theta \{J(\vec{\theta}, X)\}$. To minimize the function we use the Gradient Descent algorithm. We start with an initialization of $\vec{\theta} = \vec{\theta}_0$ and we set a learning rate η, small enough to assure convergence. We will iterate through the dataset several times (called epochs) and after each step we will update the parameters with the following rule: $\vec{\theta}_{t+1} = \vec{\theta}_t - \eta \cdot \nabla_\theta J$ were $\nabla_\theta J$ is the gradient of J with respect to θ. (Instead of the whole dataset, we can update after every batch, a part of the dataset).

```
fit function: (having matrices X, y)        X: (N, D)   y: (N,1)
    theta = random.normal (X.shape[0]+1)  #we initialize theta with small random values, with theta = [θ1 ... θD, b]
    eta = 0.1
    n_epochs = 10   # for example
    X_ext = np.hstack((1, X))   #we extend every sample x, to have 1 on top, to simplify.
    for epoch in n_epochs:
        y_pred = predict(X_ext, theta)
        J = 1/N * np.sum(y_pred - y)^2        gradient calculation before
        theta = theta - eta * 2 * ... * X_exp   #all this using matrix multiplications, if needed, and transpose to get correct dimensions
    return theta
```


Notez avec exactitude votre numéro de table ou d'inscription.
Il est interdit au candidat de signer sa copie ou d'y mettre un signe quelconque pouvant identifier l'auteur de la copie.

c) for K>2 we have the same idea, classes with K>2 will be like class number 2:

$$\sigma(\vec{w}_{1K} \cdot \vec{x}_n) - (t_{n1} - \sum_{k=2}^{K} t_{nK})$$

so $t_{n,kk'} = \begin{bmatrix} +\delta_{k1} \\ -\delta_{k2} \\ -\delta_{kK} \end{bmatrix}$ =?

NO

0

2. K>2 case, parameters in the model?

$$|\theta| = \frac{K(K-1)}{2}$$

we have K classes, and each one has a plane for all the others, but they're reciprocal.

K classes, each one a plane for the rest (K−1)

0,25 $|\theta|_{\text{might be}}$ K(K−1), but we know that $w_{ij} = -w_{ji}$,

we only have half of the parameters:

$$|\theta| = \frac{K(K-1)}{2}$$

× D !

? 2 ✓

3. 0,5 $J(\theta, x) = \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{K} \sum_{k' \neq k}^{K} [\sigma(\vec{w}_{kk'} \cdot \vec{x}_n) - t_{n,kk'}]$?

4. $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ $x: [-\infty, +\infty]$, $y: [-1, 1]$ $\frac{\partial}{\partial x} \tanh(u(x)) = \frac{4u'(x)}{(\cosh^2(u(x)))}$ $\cosh(x) = \frac{e^x + e^{-x}}{2} \geq 1, \forall x$

$\sigma(\cdot) = \tanh(\cdot)$.

a) Derivate GD for this $J(\theta, x)$

we call $(\vec{w}_{kk'} \cdot \vec{x}_n) = u(w, x)$

$$\nabla_w J = \nabla_w \frac{1}{2} \sum_{i}^{N} \sum_{k}^{K} \sum_{k'}^{K} [\sigma(u) - t_{n,kk'}]^2 = \quad \checkmark \text{ ok}$$

$$= \frac{1}{2} \sum\sum\sum 2 [\sigma(u) - t_{n,kk'}] \cdot \frac{4 \cdot u'(\cdot)}{\cosh^2(u)} = 4 \sum\sum\sum [\sigma(u) - t_{n,kk'}] \cdot \frac{u'}{\cosh^2(u)}$$

$$\nabla_w J = 4 \sum_{i}^{N} \sum_{k}^{K} \sum_{k'}^{K} [\sigma(\vec{w}_{kk'} \cdot \vec{x}_n) - t_{n,kk'}] \cdot \frac{\vec{x}_n}{\cosh^2(\vec{w}_{kk'} \cdot \vec{x})} \checkmark$$

$$w_{t+1} = w_t - \eta \frac{1}{N} 4 \sum_{i}^{N} \sum_{k}^{K} \sum_{k'}^{K} [\sigma(\vec{w}_{kk'} \cdot \vec{x}_n) - t_{n,kk'}] \cdot \frac{\vec{x}_n}{\cosh^2(\vec{w}_{kk'} \cdot \vec{x})} \checkmark$$

~~$w_{t+1} = w_t - \eta \cdot 4 \cdot$~~

0,5 $w_{t+1} = w_t - \eta \cdot 4 \cdot \sum\sum \nabla \sigma(W_{kk'}^T X) \cdot \frac{X_{N,D}}{\cosh^2(W^T X)}$ = ??

0,5 $= w_t - \eta \cdot 4 \cdot \sigma(W^T X) \cdot \frac{X_{N,D}}{\cosh^2(W^T X)}$ we apply sigma to every row

6. $\hat{y} = \text{sign}[\sigma(\vec{w}_{kk'} \cdot \vec{x}) - t_{n,kk'}]$ we will have K(K−1) predictions / half repeated

5. Nothing will happen to that plane, no parameters will be updated. ok

1

5

ANNÉE : ....

UNITÉ D'ENSEIGNEMENT : ....

ÉPREUVE DE : ....

NOTE /20	Coefficient	Note affectée du coefficient

NOM : [redacted]

Prénoms : [redacted]

N° D'INSCRIPTION OU DE TABLE

SALLE D'EXAMEN :

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

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

② LINEAR REGRESSION (II)

4(ii) c) In our second model our model is minimizing a cost function that minimizes the MAE, as a result we can expect that this model LinReg2 will perform better than LinReg that was trained to minimize squared errors. (As long as the regularization is well defined). The higher the degree of the polinomial, for a measure of error, the higher the penalty for big errors, so a LinReg2 will be better trained for big errors than LinReg, probably his error will be smaller.

0,75 ✓ thanks!

5. Mini-batch

a) Now we will iterate over each epoch but also over batches:

```
: (as before)
for epoch in n-epochs:
    batchs = split-in-batches (X-ext)
    for batch in batches:
        y_pred = predict (X, theta)
        J = cost according to the cost function decided (for each mini-batch)
        theta = theta - eta * grad_J          # Grad_J = ∇θ J
                                              # we update theta after each mini-batch
return theta.
```

1 ✓

Mathematically it means that now the cost function will be:

$$J(\theta, X) = \frac{1}{M} \sum_{n=1}^{M} (f(\vec{x}) - y)^2$$ where M is the number of samples in the minibatch.

after we would update

$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta J$ (update) ~~And we would repeat this~~

So after each iteration for a mini-batch, we would continue with the following one to do a whole epoch, and iterate again, until we reach the desired number of epochs

PARTIE A RABATTRE

b) for big datasets, as the ones used in Deep Learning, the samples might have great variance, the cost function can be very complex, have different minimums, so the gradient won't probably follow a clear descent direction, but an average above all the different ones, making convergence slower and probably leading to not so good local minima.

c) mini-batches can be also useful when we have limited computational resources; if we can't handle big amounts of data, mini-batches allow us to process the data in parts, making the task less ressource-demanding

③ 0-1-0 classification

1. k=2 a) n is class 1?

④ 1) Mushroom property a) Given features of a mushroom, say if it is edible or poisonous

b) Supervised task, classification (edible/poisonous) c) data are features, usually categorical that we will encode (probably one-hot) and some other numerical ones (height, width), each mushroom will be an array with a component for each feature. d) SVM for classification seems fine

e) we should take into account that (predicting true = poisonous) a false positive doesn't have the same consequences as a false negative. We will adapt our performance metric so our classifier has more false positive (we won't eat mushrooms that were actually edible) than false negative (we don't want to eat something that is poisonous)

2) Pokémon a) Our task is ranking Pokémon, to say which one will win. We need to create a strength metric given its features. b) we should do a supervised regression c) Data will be an array of features for each Pokémon d) we could try linear regression, but it seems a quite complicated task, so neural networks might work better.

e) There's nothing special to worry about, maybe the fact that some Pokémon won't appear as much as others in the dataset, we might have unbalanced dataset but it will probably won't have any huge impact.

3. World temperatures a) The goal is to predict future temperatures, give a prediction for temperature based on past records. b) It will be an unsupervised task, since we don't know which temperature will have in the future, but we could train our model to predict temperatures from the past or present, and then predict in the future (that part would be supervised).
c) Data is an array for each city containing the numerical features.
e) ~~we could include a periodic function to manage this period of one year.~~
we could generate new dimensions for each season, according to the time of recording and maybe latitude for the hemisphere, so each season is in a different component.

4. Footballers positions

a) We want to predict the label position given a football player's characteristics.
b) It is a supervised task of classification. c) The data is array for each player containing numerical and categorical data (that will be encoded), each one in a component. d) As a complex classification task SVM can work well, or even Neural Networks.
e) To do this dimensional reduction, we need to cluster these labels, create groups, for this we can use an unsupervised method like K-means. This way, instead of 27 different labels, we can group our players in the desired number of different labels (9-12). Afterwards, we could use and predict these new labels.

③ 0-1-0 classification

1. $K=2$ $\sigma(\vec{w}_{12}\vec{x}_n) - (t_{n1} - t_{n2})$

a) $n=1$? $\sigma(K^+) - \left(\begin{bmatrix}1\\0\end{bmatrix} - \begin{bmatrix}0\\0\end{bmatrix}\right) = K - \begin{bmatrix}1\\0\end{bmatrix}$

$n=2$? $\sigma(K^+) - \left(\begin{bmatrix}0\\0\end{bmatrix} - \begin{bmatrix}0\\1\end{bmatrix}\right) = K + \begin{bmatrix}0\\1\end{bmatrix} = K - \begin{bmatrix}0\\-1\end{bmatrix}$

We have $\sigma(\vec{w}_{12} \cdot \vec{x}) - \vec{t}_{n,12}$, where $\vec{t}_{n,12} = \begin{bmatrix}1 \text{ if } n=1\\ -1 \text{ if } n=2\end{bmatrix}$,

so we need $\vec{v}$ to be: $\begin{bmatrix}\delta_{n,1}\\ -\delta_{n,2}\end{bmatrix}$

b) $\vec{w}_{12}$ is $(-1)\cdot\vec{w}_{21}$, is the same plane, but pointing in opposite directions to have $\vec{w}_{12}\cdot x > 0$ if x is class one (< 0 if class two) and $\vec{w}_{21}\cdot x > 0$ if x is class 2 (< 0 if class 1)

$t_{n,12}$ is also $(-1)\cdot t_{n,21}$