Machine Learning

Homework 4

(1) Monotonicity of loss and regularizer as the regularization parameter changes.

Chose $\theta \in \mathbb{R}^m$ to minimize regularized empirical risk, $L(\theta) + \lambda r(\theta)$

$\hat{\theta}$

Intuition: As λ increases, $r(\theta)$ decreases while $L(\theta)$ increases.

Suppose, $0 < \lambda < \tilde{\lambda}$. Let θ^* minimize $L(\theta) + \lambda r(\theta)$

$\tilde{\theta}^*$ minimize $L(\theta) + \tilde{\lambda} r(\theta)$

(a) Show $r(\theta^*) \geq r(\tilde{\theta}^*)$.

Given θ^* is minimizer of $L(\theta) + \lambda r(\theta)$

$\therefore L(\theta) + \lambda r(\theta) \geq L(\theta^*) + \lambda r(\theta^*)$

Similarly for any $\tilde{\theta}^*$ we have,

$$L(\tilde{\theta}^*) + \lambda r(\tilde{\theta}^*) \geq L(\theta^*) + \lambda r(\theta^*) \longrightarrow (1)$$

Also given, $\tilde{\theta}^*$ is minimizer of $L(\theta) + \tilde{\lambda} r(\theta)$

$\therefore L(\theta) + \tilde{\lambda} r(\theta) \geq L(\tilde{\theta}^*) + \tilde{\lambda} r(\tilde{\theta}^*)$

Similarly for any θ^* we have,

$$L(\theta^*) + \tilde{\lambda} r(\theta^*) \geq L(\tilde{\theta}^*) + \tilde{\lambda} r(\tilde{\theta}^*) \longrightarrow (2)$$

Adding (1) and (2),

$$\cancel{L(\tilde{\theta}^*)} + \lambda r(\tilde{\theta}^*) + \cancel{L(\theta^*)} + \tilde{\lambda} r(\theta^*) \geq \cancel{L(\theta^*)} + \lambda r(\theta^*) + \cancel{L(\tilde{\theta}^*)} + \tilde{\lambda} r(\tilde{\theta}^*)$$

$$(\tilde{\lambda} - \lambda)\ r(\theta^*) \geq (\tilde{\lambda} - \lambda)\ r(\tilde{\theta}^*)$$

Since $\tilde{\lambda} > \lambda$ $\quad \tilde{\lambda} - \lambda > 0$

$\therefore\ r(\theta^*) \geq r(\tilde{\theta}^*)$.

$\therefore$ Increasing λ will never make our regularization error larger.

(b) Show $L(\theta^*) \leq L(\tilde{\theta}^*)$

→ from (a), equation (1) and (2)

(1) $\quad L(\tilde{\theta}^*) + \lambda r(\tilde{\theta}^*) \geq L(\theta^*) + \lambda r(\theta^*) \quad \times \tilde{\lambda}$

(2) $\quad L(\theta^*) + \tilde{\lambda} r(\theta^*) \geq L(\tilde{\theta}^*) + \tilde{\lambda} r(\tilde{\theta}^*) \quad \times \lambda$

multiplying (1) with $\tilde{\lambda}$ and (2) with λ and adding, we get,

$$\tilde{\lambda} L(\tilde{\theta}^*) + \lambda L(\theta^*) \geq \tilde{\lambda} L(\theta^*) + \lambda L(\tilde{\theta}^*)$$

Rearranging,

$$(\tilde{\lambda} - \lambda) L(\tilde{\theta}^*) \geq (\tilde{\lambda} - \lambda) L(\theta^*)$$

Since $\tilde{\lambda} > \lambda$, $\tilde{\lambda} - \lambda > 0$.

$$\therefore L(\tilde{\theta}^*) \geq L(\theta^*)$$

$\therefore$ Increasing λ will never decrease our training error.

(2) MAP interpretation of regularized empirical losso minimization.

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{n} -\log p(y_i | x_i ; \theta) - \log p(\theta)$$

Given MAP formulation for estimating θ.

(a) $p(y|x, \theta) \sim N(\phi^T \theta, \sigma^2)$ and $p(\theta) \sim N(0, \sigma_0^2 I)$

→ log likelihood estimate of $p(y|x, \theta)$,

$$p(y|x, \theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(y_i - \phi_i^T\theta)^2\right)$$

$$\log p(y|x, \theta) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \phi_i^T\theta)^2$$

$$= \underbrace{-\frac{n}{2}\log(2\pi\sigma^2)}_{\text{constant}} - \frac{1}{2\sigma^2}(y - \phi^T\theta)^2 \quad //$$

Log likelihood estimate of $p(\theta)$,

$$\log p(\theta) = \underbrace{\frac{m}{2}\log(2\pi\sigma_0^2)}_{\text{constant}} - \frac{1}{2\sigma_0^2}\|\theta\|^2$$

Ignoring constant ~~terms~~ terms and in terms of the given ~~MAP~~ MAP formulation

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{n} (y_i - \phi_i^T \theta)^2 + \frac{\sigma^2}{\sigma_0^2} \|\theta\|^2 \longrightarrow \Rightarrow \text{Ridge Regression}$$

In terms of empirical risk formulation,

$$\boxed{\underset{\theta}{\text{minimize}} \frac{1}{n} \sum_{i=1}^{n} l(\phi_i^T \theta, y_i) + \lambda r(\theta)} \longrightarrow (1)$$

$$\boxed{\therefore \lambda = \frac{\sigma^2}{n\sigma_0^2}}$$

(b) $p(y|x,\theta) \sim N(\phi^T\theta, \sigma^2)$ and $p(\theta)$ follows a multivariate Laplacian distribution:

$$p(\theta) = \prod_{j=1}^{m} \frac{1}{2a} \exp\left(-\frac{|\theta_j|}{a}\right)$$

$\rightarrow$ from (a), $\log p(y|x;\theta) = \frac{-n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(y - \phi^T\theta)^2$

$$\log p(\theta) = -m \log(2a) - \sum_{j=1}^{m} \frac{|\theta_j|}{a}$$

$$= - m\log(2a) - \frac{1}{a} \|\theta\|_1$$

In terms of MAP,

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{n} (y_i - \phi_i^T\theta)^2 + \frac{\sigma^2}{a} \|\theta\|_1$$

$\Rightarrow$ Lasso Regression

$\therefore$ In terms of (1) we get,

$$\boxed{\lambda = \frac{\sigma^2}{na}}$$

(d) $p(y|x,\theta) = \dfrac{1}{1+\exp(-y\phi^T\theta)}$ where $y = \pm 1$ and $p(\theta)$ as in (6)

$\longrightarrow$ $\log p(y|x,\theta) = -\log(1+\exp(-y\phi^T\theta))$

log likelihood estimate, of $p(y|x;\theta)$

log likelihood estimate of $p(\theta)$ from (6),

$$\log(p(\theta)) = -m\log(2a) - \frac{1}{a}\|\theta\|_1$$

$\therefore$ In terms of MAP formulation,

$$\underset{\theta}{\text{minimize}} \quad \sum_{i=1}^{n} \log(1+\exp(-y\phi^T\theta)) + \frac{1}{a}\|\theta\|_1.$$

L1-regularized logistic regression.

$$\boxed{\lambda = \frac{1}{na}}$$

Using common empirical loss minimization formulation,

(c) $p(y|x,\theta) = Pr[yu \geq 0]$ where $y = \pm 1$, $p(u|x,\theta) \sim N(\phi^T\theta, \sigma_0^2)$

and $p(\theta) \sim N(0, \sigma_0^2 I)$

$\rightarrow$

$$p(y|x,\theta) = \int_0^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(u - y\phi^T\theta)^2\right) du$$

$$= \int_{-\infty}^{y\phi^T\theta} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\cdot u^2\right) du$$

$=$

$$\therefore \log p(y|x,\theta) = \log \int_{-\infty}^{y\phi^T\theta} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du$$

$\log p(y|x,\theta) =$

$$\log \Phi(y\phi^T\theta).$$

Φ is cummulative distribution function,

$$\Phi(\theta) = \int_{-\infty}^{\theta} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du$$

~~plot~~ ~~do~~ $\log(p(\theta))$ from (a),

$$\log(p(\theta)) = \frac{m}{2}\log(2\pi\sigma_0)^2 - \frac{1}{2\sigma_0^2}\|\theta\|^2$$

$\therefore$ In terms of MAP formulation,

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{n} -\log \Phi(y\phi^T\theta) + \frac{1}{\sigma_0^2}\|\theta\|^2$$

$$\boxed{\therefore \lambda = \frac{1}{n\sigma_0^2}}$$

By using common empirical loss minimization,

$$\underset{\theta}{\text{minimize}} \ \frac{1}{n}\sum_{i=1}^{n} l_i(\theta) + \lambda \phi r_i(\theta)$$

L2-regularized Probit Regression:

(3) Proximal operator for group lasso regularizer

(a) Subdifferential of function $\|\theta_j\|$

→

$$\|\theta_j\| = \sqrt{\sum_{c=1}^{k} \theta_{jc}^2}$$

Euclidean norm is not differentiable at 0,

$\therefore$ lets consider $\neq 0$ case,

If $\theta_j \neq 0$, partial differentiation is given as,

$$\frac{\partial \|\theta_j\|}{\partial \theta_{jc}} = \frac{\theta_{jc}}{\|\theta_j\|}$$

$$\therefore \partial \|\theta_j\| = \frac{\theta_j}{\|\theta_j\|} \ //$$

Now for, $\theta_j = 0$,

The $\|x\|$ can be written as $\max_{\|a\|=1} a^T x$

when $x \neq 0$, the maximum is attained when $a = \frac{x}{\|x\|}$

and when $x=0$, the max is attained for any a with $\|a\|=1$.

$\therefore$ Any vector in unit ball ($\|a\| \leq 1$) is valid subgradient.

$$\boxed{\therefore\ \partial\|\theta_j\| = \begin{cases} \left\{ \dfrac{\theta_j}{\|\theta_j\|} \right\}, & \theta_j \neq 0 \\ \{ a \mid \|a\| \leq 1 \}, & \theta_j = 0 \end{cases}}$$

(b) Derive the solution of

$$\underset{\theta_j}{\text{minimize}}\ \rho\|\theta_j\| + \frac{1}{2}\|\theta_j - \tilde{\theta}_j\|^2$$

$\longrightarrow$ Gradient for proximal term $\left(\frac{1}{2}\right)\|\theta_j - \tilde{\theta}_j\|^2$ is $\theta_j - \tilde{\theta}_j$

Equating differential to 0, since optimal,

$$\partial\rho\|\theta_j\| + (\theta_j - \tilde{\theta}_j) = 0$$

$$\partial\|\theta_j\| = \frac{1}{\rho}(\tilde{\theta}_j - \theta_j) \longrightarrow (1)$$

We know,

$$\partial\|\theta_j\| = \begin{cases} \dfrac{\theta_j}{\|\theta_j\|}, & \theta_j \neq 0 \\ \{ a \mid \|a\| \leq 1, & \theta_j = 0 \end{cases} \quad \text{from (a)}$$

If $\theta_j = 0$, (1) become,

$$1 \geq \left\|\left(\frac{1}{\rho}\right)\tilde{\theta}_j\right\| \Rightarrow \underline{\rho \geq \|\tilde{\theta}_j\|}$$

$$\underline{\theta_j = 0}$$

For $\theta_j \neq 0$, (1) becomes

$$0 = \rho \cdot \frac{\theta_j}{\|\theta_j\|} + (\theta_j - \tilde{\theta}_j)$$

$$\tilde{\theta}_j = \theta_j \underbrace{\left(\frac{\rho}{\|\theta_j\|} + 1\right)}_{\text{constant}}$$

$$\therefore \theta_j = \alpha \tilde{\theta}_j \longrightarrow (2)$$

That is optimal θ_j is in a direction of $\tilde{\theta}_j$

Substituting (2),

$$\cancel{\tilde{\theta}_j} = \alpha \cancel{\tilde{\theta}_j} \left(\frac{\rho}{\alpha \|\tilde{\theta}_j\|} + 1\right)$$

$$1 = \frac{\alpha \rho}{\alpha \|\tilde{\theta}_j\|} + \alpha$$

$$\alpha = 1 - \frac{\rho}{\|\tilde{\theta}_j\|}$$ putting α in (2)

$$\theta_j = \left(1 - \frac{\rho}{\|\tilde{\theta}_j\|}\right)\tilde{\theta}_j \quad \text{given } \|\tilde{\theta}_j\| > \rho$$

$$\therefore \text{Prox}_{\rho\|\cdot\|}(\tilde{\theta}_j) = \begin{cases} 0, & \|\tilde{\theta}_j\| \le \rho \\ \left(1 - \frac{\rho}{\|\tilde{\theta}_j\|}\right)\tilde{\theta}_j & \|\tilde{\theta}_j\| > \rho \end{cases}$$

(4) News articles classification

$$\underset{\theta_1, \dots, \theta_k}{\text{minimize}} \quad \frac{1}{n} \sum_{i=1}^{n} \max_c \left(x_i^T \theta_c - x_i^T \theta_{y_i} + 1_{y_i \neq c} \right) + \lambda \sum_{j=1}^{m} \sqrt{\sum_{c=1}^{k} \theta_{jc}^2}$$

(a) Derive stochastic proximal subgradient algorithm for solving it.

→ ~~Recon~~

$$l_i(\theta) = \cancel{\max} \max_c \left(x_i^T \theta_c - x_i^T \theta_{y_i} + 1_{y_i \neq c} \right)$$

for a particular sample x_i,
~~let's say~~ Since only one class attains maximum, the function is differentiable.

$$\nabla l_i(\theta) = 0$$

$$\nabla_{\theta_{y_i}} l(\theta^{(t)}) = -x_i \quad \text{corresponding to correct class}$$

$$\nabla_{\theta_c} l(\theta^{(t)}) = x_i \quad \text{corresponding to incorrect class}$$

The intuition is that gradient descent update at most two columns of Θ matrix, one for correct label and another for predicted label.

$$\nabla_{\theta_c} l_i(\theta^{(t)}) = \begin{cases} x_i \,, & c = \hat{y}_i \\ -x_i \,, & c = y_i \\ 0 \,, & \text{otherwise} \end{cases}$$

Apply row-wise block soft-thresholding on Θ. The norm of each row is calculated and if it's ~~a~~ less than ~~γ~~ $\gamma^{(t)}\lambda$ then entire row is zero. Otherwise row decreases by $\gamma^{(t)}\lambda$

$$\theta_{y_i}^{(t+1)} = \theta_{y_i}^{(t)} + \gamma^{(t)} x_i \Rightarrow \text{For incorrect}$$

$$\theta_{y_i}^{(t+1)} = \theta_{y_i}^{(t)} - \gamma^{(t)} x_i \Rightarrow \text{For correct}$$

New θ for proximal operation,

$$\theta^{t+1} = \theta^{t} - \lambda \nabla L$$

Subgradient is calculated by ~~that~~ above method

$$\Rightarrow \theta_j^{(t+1)} = \begin{cases} 0 & \text{if } v \le \gamma^{(t)}\lambda \\ \left(1 - \frac{\gamma^{(t)}\lambda}{v}\right)\theta_j^{t+1} & \text{otherwise} \end{cases}$$

here v is the current row / row wise norm of $\Theta^{(t+1)}$.

$$v = \|\theta_j^{(t+1)}\|$$

(d) As λ increases the group sparsity is more and more rows are zeroed out.

There fore more ignored words.

Question 4 Attempt 2

April 22, 2025

[1]:
```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
from sklearn.naive_bayes import MultinomialNB as MNB
from sklearn.metrics import accuracy_score
import math
import numpy as np
from numpy.linalg import qr, norm
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture
import matplotlib.cm as cm
import urllib
```

[75]:
```
X_train_input = pd.read_csv("20news-bydate/matlab/train.data", delimiter="␣
↪",names = ["docIdx", "wordIdx", "freq"],)
y_train_input = pd.read_csv("20news-bydate/matlab/train.label",␣
↪names=["labels"])
X_test_input = pd.read_csv("20news-bydate/matlab/test.data", delimiter=" ",␣
↪names = ["docIdx", "wordIdx", "freq"],)
y_test_input = pd.read_csv("20news-bydate/matlab/test.label", names=["labels"])

X_train = X_train_input.to_numpy()
y_train = y_train_input.to_numpy()
X_test = X_test_input.to_numpy()
y_test = y_test_input.to_numpy()

samples = max(X_train[:,0])
test_samples = max(X_test[:,0])
vocab = len(np.unique(np.concatenate((X_train[:, 1], X_test[:, 1]))))
train_X = np.zeros((samples, vocab))
test_X = np.zeros((test_samples, vocab))

for i in X_train:
    train_X[i[0]-1,i[1]-1] = i[2]

for i in X_test:
    test_X[i[0]-1,i[1]-1] = i[2]
```

```
k = int(max(y_train[:, 0]))
n,m = train_X.shape
```

[79]:
```
data = fetch_20newsgroups(subset='all', remove=('headers','footers','quotes'))
vec  = CountVectorizer(max_features=2000)
X = vec.fit_transform(data.data).toarray()  # shape (N, m)
y = data.target
feat_names = vec.get_feature_names_out()

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=0
)
n, m = X_train.shape
k      = np.unique(y).size
```

[83]:
```
def stochastic_proximal_gradient_descent_algo(num_iters, lambda_,␣
 ↪step_size_type="diminishing"):
    theta = np.zeros((m,k))
    ret_accuracy = []
    for t in range(1,num_iters):
        if step_size_type == "diminishing":
            step_size = 1.0/t
        else:
            step_size = 0.01
        i = np.random.randint(n)
        x_i = X_train[i,:]
        y_i = y_train[i]

        conditional_y = np.ones(k, dtype=int)
        conditional_y[y_i] = 0


        y_i_hat = np.argmax( x_i @ theta - x_i @ theta[:,y_i] + conditional_y)
        theta_temp = theta

        # print(theta_temp.shape)
        # print(theta.shape)
        # # print(x_i.shape)

        if y_i != y_i_hat:
            theta_temp[:, y_i] = theta[:, y_i] + step_size * x_i
            theta_temp[:, y_i_hat] = theta[:, y_i_hat] - step_size * x_i

        for j in range(m):
            v = np.linalg.norm(theta_temp[j])
            if v <= step_size * lambda_ :
```

```
                theta_temp[j] = 0
            else:
                theta_temp[j] = (1 - ((step_size * lambda_) / v) ) *␣
↪theta_temp[j]

        theta = theta_temp

        if t%1000 == 0:
            print(t, end=" ")
            ret_accuracy.append(get_prediction_accuracy(theta))

    # print()
    return ret_accuracy, theta
```

[86]:
```
def get_prediction_accuracy(theta):
    y_pred = np.argmax(X_test @ theta, axis = 1)
    return accuracy_score(y_test, y_pred)
```

[90]:
```
lambdas = [10, 1, 0.1, 0.01]
results = {}
thetas = {}
for i in lambdas:
    print("Lambda =", i)
    results[i], thetas[i] = stochastic_proximal_gradient_descent_algo(100001,i)
```

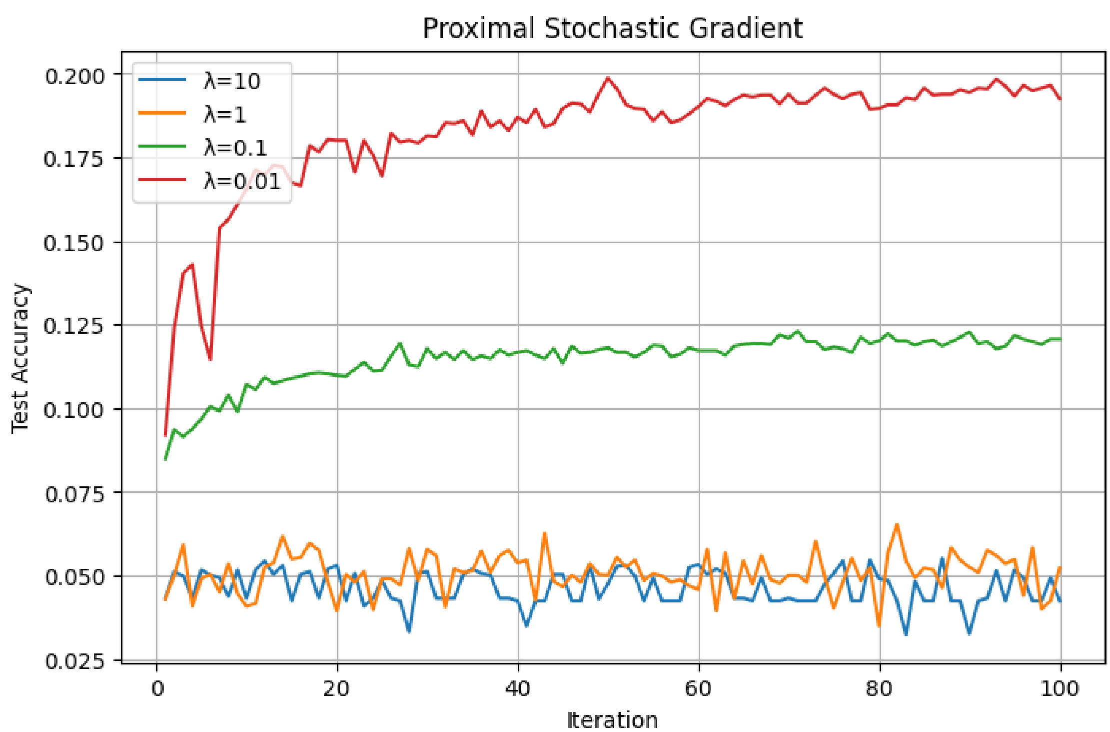
```
Lambda = 10
1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000 13000 14000 15000
16000 17000 18000 19000 20000 21000 22000 23000 24000 25000 26000 27000 28000
29000 30000 31000 32000 33000 34000 35000 36000 37000 38000 39000 40000 41000
42000 43000 44000 45000 46000 47000 48000 49000 50000 51000 52000 53000 54000
55000 56000 57000 58000 59000 60000 61000 62000 63000 64000 65000 66000 67000
68000 69000 70000 71000 72000 73000 74000 75000 76000 77000 78000 79000 80000
81000 82000 83000 84000 85000 86000 87000 88000 89000 90000 91000 92000 93000
94000 95000 96000 97000 98000 99000 100000 Lambda = 1
1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000 13000 14000 15000
16000 17000 18000 19000 20000 21000 22000 23000 24000 25000 26000 27000 28000
29000 30000 31000 32000 33000 34000 35000 36000 37000 38000 39000 40000 41000
42000 43000 44000 45000 46000 47000 48000 49000 50000 51000 52000 53000 54000
55000 56000 57000 58000 59000 60000 61000 62000 63000 64000 65000 66000 67000
68000 69000 70000 71000 72000 73000 74000 75000 76000 77000 78000 79000 80000
81000 82000 83000 84000 85000 86000 87000 88000 89000 90000 91000 92000 93000
94000 95000 96000 97000 98000 99000 100000 Lambda = 0.1
1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000 13000 14000 15000
16000 17000 18000 19000 20000 21000 22000 23000 24000 25000 26000 27000 28000
29000 30000 31000 32000 33000 34000 35000 36000 37000 38000 39000 40000 41000
42000 43000 44000 45000 46000 47000 48000 49000 50000 51000 52000 53000 54000
55000 56000 57000 58000 59000 60000 61000 62000 63000 64000 65000 66000 67000
68000 69000 70000 71000 72000 73000 74000 75000 76000 77000 78000 79000 80000
```

```
81000 82000 83000 84000 85000 86000 87000 88000 89000 90000 91000 92000 93000
94000 95000 96000 97000 98000 99000 100000 Lambda = 0.01
1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000 13000 14000 15000
16000 17000 18000 19000 20000 21000 22000 23000 24000 25000 26000 27000 28000
29000 30000 31000 32000 33000 34000 35000 36000 37000 38000 39000 40000 41000
42000 43000 44000 45000 46000 47000 48000 49000 50000 51000 52000 53000 54000
55000 56000 57000 58000 59000 60000 61000 62000 63000 64000 65000 66000 67000
68000 69000 70000 71000 72000 73000 74000 75000 76000 77000 78000 79000 80000
81000 82000 83000 84000 85000 86000 87000 88000 89000 90000 91000 92000 93000
94000 95000 96000 97000 98000 99000 100000
```

[91]:

```python
plt.figure(figsize=(8,5))
for res in results:
    plt.plot([i for i in range(1,101)], results[res], label=f" ={res}")

plt.xlabel("Iteration x1000")
plt.ylabel("Test Accuracy")
plt.title("Proximal Stochastic Gradient")
plt.legend()
plt.grid(True)
plt.show()
```



[95]:
```
dropped = {}
for res, th in thetas.items():
    norms = np.linalg.norm(th, axis=1)
    zero_idx = np.where(norms < 1e-6)[0]
    dropped[res] = feat_names[zero_idx]
    print(f"\n ={res}: dropped {len(dropped[res])} terms")
    # print(dropped.tolist())
```

```
=10: dropped 2000 terms

=1: dropped 1851 terms

=0.1: dropped 1323 terms

=0.01: dropped 163 terms
```

[99]:
```
for res in thetas:
    print(f"\n ={res}")
    print(dropped[res].tolist()[:100])
```

```
=10
['00', '000', '01', '02', '03', '04', '05', '06', '0d', '0t', '10', '100', '11',
'12', '128', '13', '14', '145', '15', '150', '16', '17', '18', '19', '1988',
'1989', '1990', '1991', '1992', '1993', '1d9', '1st', '1t', '20', '200', '21',
'22', '23', '24', '25', '250', '256', '26', '27', '28', '29', '2di', '2nd',
'2tm', '30', '300', '31', '32', '33', '34', '34u', '35', '36', '37', '38',
'386', '39', '3d', '3rd', '3t', '40', '400', '41', '42', '43', '44', '45', '46',
'47', '48', '486', '49', '50', '500', '51', '52', '53', '54', '55', '56', '57',
'58', '59', '60', '61', '63', '64', '65', '66', '68', '6ei', '6um', '70', '71',
'72']

=1
['00', '000', '01', '02', '03', '04', '05', '06', '0d', '0t', '100', '11',
'128', '13', '14', '145', '15', '150', '16', '17', '18', '19', '1988', '1989',
'1991', '1992', '1993', '1d9', '1st', '1t', '200', '21', '22', '23', '24', '25',
'250', '256', '26', '27', '28', '29', '2di', '2nd', '2tm', '30', '300', '31',
'32', '33', '34', '34u', '35', '36', '37', '38', '386', '39', '3d', '3rd', '3t',
'400', '41', '42', '43', '44', '45', '46', '47', '48', '486', '49', '50', '500',
'51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '63', '64',
'65', '66', '68', '6ei', '6um', '70', '71', '72', '73', '75', '75u', '76', '77']

=0.1
['01', '02', '03', '04', '05', '0d', '0t', '128', '145', '150', '1d9', '1st',
'1t', '256', '2di', '2nd', '2tm', '300', '34u', '35', '37', '386', '39', '3d',
'3rd', '3t', '40', '400', '42', '45', '46', '47', '48', '486', '500', '51',
'52', '53', '54', '55', '56', '57', '58', '59', '60', '63', '64', '65', '6ei',
```

```
'6um', '70', '72', '73', '75', '75u', '76', '77', '78', '7ey', '7u', '80',
'800', '82', '84', '85', '86', '88', '90', '91', '92', '95', '99', '9v', '__',
'a86', 'ability', 'able', 'absolutely', 'ac', 'access', 'according', 'account',
'across', 'act', 'actions', 'active', 'activities', 'activity', 'acts',
'actual', 'actually', 'add', 'added', 'additional', 'address', 'adl',
'administration', 'advantage', 'advice', 'afraid']

 =0.01
['06', '0d', '1988', '1989', '1d9', '1st', '2di', '2nd', '2tm', '34u', '36',
'37', '39', '3t', '43', '49', '51', '53', '57', '59', '68', '6um', '70', '72',
'73', '76', '77', '78', '7ey', '84', '9v', 'added', 'advice', 'agencies',
'agents', 'aids', 'alive', 'alternative', 'america', 'army', 'attention',
'audio', 'azerbaijan', 'believed', 'bj', 'block', 'brought', 'bxn', 'carried',
'cc', 'choose', 'chz', 'commercial', 'committed', 'committee', 'count', 'd9',
'description', 'determine', 'directly', 'disagree', 'document', 'draw', 'edge',
'education', 'entire', 'expected', 'eye', 'eyes', 'feet', 'fi', 'g9v', 'george',
'gk', 'helps', 'hiv', 'homosexuality', 'independent', 'istanbul', 'jobs',
'justice', 'lead', 'leading', 'learn', 'length', 'listed', 'lk', 'location',
'lose', 'm3', 'march', 'md', 'mq', 'mt', 'mv', 'mw', 'myers', 'nation',
'notice', 'nuclear']
```

[ ]: