Classification Problem Formulation

Dr. Kalidas Y., IIT Tirupati

By the end of this lecture, you will understand what is a classification problem and be able to formulate simple ones

Loss function

$$L(w) = -logP(D|w) = -\sum_{(x,y)\in D} (y * (w \cdot x) - log(1 + e^{w \cdot x}))$$

44) key phrase… “Classification Problem”

- Category

- Examples
  - Cat or Not Cat
  - Good or Bad
  - Rainy Day, Cloudy Day, Bright Day, Warm Day, Hot Day
  - Tall, Moderately Tall, Short
  - Etc.

45) key phrase… “Binary Classification”

- Two Categories

- Examples
  - Cat or Not Cat
  - Good or Bad

46) key phrase… “Multi-class Classification”

- Multiple Categories

- Examples
  - Fault categories – Valve fault, Temperature fault, Pressure fault
  - Identify of a person – you may have thousands of people and millions of photos
  - Common objects – Car, Bus, Cycle, Person, Animal etc.

47) key phrase… “Threshold”

- Score
- Threshold
- Above it → One class
- Below it → Another class
- *Craft loss function accordingly*

48) key phrase… “SoftMax”

- K scores (*assume some numbers smoothed out in some way for now*)
- Pick maximal score – for e.g. i^{th} is maximum
- Select i^{th} class
- *Craft loss function accordingly*

Example 1 – Logistic Loss Function

- Knowledge of x_i (some k dimensional vector)
- Knowledge of $y_i \in \{-1,1\}$
- Model… $y = f(x) = w \cdot x$
- Loss function, $L(w) = \sum_{i=1}^{i=N} \log(1 + e^{-y_i \times w \cdot x_i})$
- Data set, $\{(x_1, y_1), \ldots, (x_N, y_N)\}$

- What happens when y_i and $\mathrm{w} \cdot x_i$ have same sign?
- What happens when y_i and $\mathrm{w} \cdot x_i$ have opposite sign?

Example 2 – Hinge Loss Function

- Knowledge of x_i (some k dimensional vector)
- Knowledge of $y_i \in \{-1,1\}$
- Model… $y = f(x) = w \cdot x$
- Loss function, $L(w) = \frac{||w||_2^2}{2} + \sum_{i=1}^{i=N} \max(0, 1 - y_i \times w \cdot x_i)$
- Data set, $\{(x_1, y_1), \ldots, (x_N, y_N)\}$

- What happens when y_i and w $\cdot\, x_i$ have same sign?
- What happens when y_i and w $\cdot\, x_i$ have opposite sign?

control amount
of regularization

ground truth

$$E(w) = \frac{1}{N} \sum_{i=1}^{i=N} L(y_i, f(x_i)) + \alpha\, R(w)$$

Regularized loss function

model

regularization
e.g. L1, L2

Loss function

general practice [optional]
to avoid numerical precision issues

REF URL – https://scikit-learn.org/stable/modules/sgd.html#sgd-mathematical-formulation

Popular Loss Functions

1. Hinge Loss: $L(y_i, f(x_i)) = \max(0{,}1 - y_i \times f(x_i))$
2. Perceptron Loss: $L(y_i, f(x_i)) = \max(0, -y_i \times f(x_i))$
3. Huber Loss: $L\big(y_i, f(x_i)\big) = \epsilon|y_i - f(x_i)| - \frac{1}{2}\epsilon^2$
4. Modified Huber Loss: $L\big(y_i, f(x_i)\big) = \max\big(0{,}1 - y_i \times f(x_i)\big)^2$
5. Logistic Loss: $L(y_i, f(x_i)) = \log(1 + e^{-y_i \times f(x_i)})$
6. Least Squares Loss: $L\big(y_i, f(x_i)\big) = \frac{1}{2}\big(y_i - f(x_i)\big)$
7. Epsilon-Insensitive Loss: $L\big(y_i, f(x_i)\big) = \max(0, |y_i - f(x_i)| - \epsilon)$

- Regularization term
    1. Lasso Regularization: $R(w) = |w|_1$
    2. Ridge Regularization: $R(w) = |w|_2^2$
    3. Elastic Net Regularization: $R(w) = \beta\ |w|_2^2 + (1 - \beta)|w|_1^2$

REF - https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

In [43]:
```
import numpy as np
from sklearn.linear_model import SGDClassifier
```

In [44]:
```
X = np.array([[-1, -1], [-2, -1], [1, 1], [2, 1]])
Y = np.array([1, 1, 2, 2])
```

In [45]:
```
clf = SGDClassifier(loss='log', fit_intercept=False) #try other values 'hinge', 'squared_loss'
```

In [46]:
```
clf.fit(X,Y)
```

Out[46]:
```
SGDClassifier(alpha=0.0001, average=False, class_weight=None, epsilon=0.1,
       eta0=0.0, fit_intercept=False, l1_ratio=0.15,
       learning_rate='optimal', loss='log', max_iter=5, n_iter=None,
       n_jobs=1, penalty='l2', power_t=0.5, random_state=None,
       shuffle=True, tol=None, verbose=0, warm_start=False)
```

In [47]:
```
print(clf.predict([[-0.8, -1]]))
```

```
[1]
```

In [48]:
```
X2 = X + 100
```

In [49]:
```
clf.fit(X2,Y)
```

Out[49]:
```
SGDClassifier(alpha=0.0001, average=False, class_weight=None, epsilon=0.1,
       eta0=0.0, fit_intercept=False, l1_ratio=0.15,
       learning_rate='optimal', loss='log', max_iter=5, n_iter=None,
       n_jobs=1, penalty='l2', power_t=0.5, random_state=None,
       shuffle=True, tol=None, verbose=0, warm_start=False)
```

In [50]:
```
X_test = np.array([[-0.8,1]]) + 100
print(clf.predict(X_test))
```

```
[1]
```

When to choose what loss function?

It is a hyper parameter as well…

We will see a bit more about these in the subsequent lecture!

49) key phrase… "hyper parameters & search"

- Each and every parameter that a human can adjust
- Those parameters which cannot be learned from data automatically
- Those knobs that require human-in-loop way of adjustment

- Example
  - 1) Degree of a polynomial in case of polynomial fitting problem
  - 2) Type of regularization to use – L1 or L2 or Elastic Net
  - 3) The multiplication factor for regularization term
  - 4) Type of loss function to use – you have about 7 popular types and even more…
  - 5) In case of tree based methods… number of trees, depth of each tree etc. (will see)

For all those knobs in your hand… try different values.. *a.k.a GRID SEARCH*

Auto-ML software can be used here.

50) key phrase… "Grid Search"

- Programatically exploring possible value settings for "hyper parameters"
- to choose the best one
- Best one in terms of the highest scoring
- Or least error
- That setting which gives highest score or least error