Machine Learning Notes

Ziwei

July 7, 2018

Contents

1 Logistic Regression

1.1 Three linear models

1. linear classification(perceptron): $h(x) = sign(s)$
   - output is 1 or 0
2. linear regression: $h(x) = s$
   - output is linear transformation
3. logistic regression: $h(x) = \theta(s)$
   - output is a probability given by a sigmoid function

where $s = \sum_{i=0}^{d} w_i x_i$

1.2 Sigmoid function

also known as the sigmoid or logistic function

$$\sigma(w^T x) = \frac{1}{1 + e^{-w^T x}}$$

1.3 Logistic Regression Function

The logistic regression model specifies the probability of a binary output $y_i \in \{0, 1\}$ given the input x_i as follows:

$$\begin{aligned} p(\mathbf{y}|\mathbf{X}, \theta) &= \prod_{i=1}^{n} \text{Ber}\left(y_i | \,\text{sigm}\left(\mathbf{x}_i \theta\right)\right) \\ &= \prod_{i=1}^{n} \left[\frac{1}{1 + e^{-x_i \theta}}\right]^{y_i} \left[1 - \frac{1}{1 + e^{-x_i \theta}}\right]^{1 - y_i} \end{aligned}$$

where $x_i \theta = \theta_0 + \sum_{j=1}^{d} \theta_j x_{ij}$

1.4 Error measure(Cross-entropy Error)

Based on **Likelihood**: if hypothesis h = f, how likely to get y from x. Given a set of training data points $(x_i, y_i), i = 1,\ldots,n$ where $y_i \in \{0, 1\}$. We need to find a weight vector w s.t. the probability of the correct y_i for x_i is high for $i = 1, ...n$

$$maxP(y = y_i | x_i; w)$$

(maxmize the log likelihood) Equiv to

$$min - \sum_{i=1}^{n} logP(y = y_i | x_i; w)$$

(minimize the negative log likelihood)

1.5 Gradient Descent

Compared to linear regression, logistic regression does not have a closed-form solution, instead of a **iterative solution** is used, which is called **gradient descent**

1. Start at w(0)
2. Takes a step along the steepest slope
3. Takes a step toward that direction
4. Repeat until no local improvement is possible

1.6 Algorithm

This is the algorithm for batch learning of logistic regression. It is very similar to linear classification's algorithm (perceptron). Both learns a linear decision boundary.

Given: $(x_i, y_i), i = 1, ...n$
Initialize $w = (0, ..., 0)$
repeat
  $\Delta = (0, ..., 0)$
  for $i = 1, ..., n$ **do**
    $\hat{y}_i = \frac{1}{1+e^{-w}T_{x_i}}$
    $\nabla = \nabla + (\hat{y}_i - y_i)\, x_i$
  end for
  $w = w - \eta\Delta$
until $|\Delta| \leq \epsilon$

1.7 Termination Criteria Considerations

The setting of terminating condition can be tricky for gradient descent

- If terminates prematurely, the algorithm may not reach the global minimum
- If there is a local minimum, the algorithm may get stuck in local minimum
- If set an expected global minimum, it may never be reached by the algorithm

1.8 Date snooping

Looking at the data before choosing the model is problematic, can lead to a fallacy

- this is different from using human expertise knowledge for feature engineering, which can help the model

2 Support Vector Machine

One of the most successful classification algorithm in machine learning. Given multiple decision boundaries that split the data, SVM seeks to find a **hyperplane** that separate the data, such that the **margin** is maximized to the nearest trained data points. It is a constrained optimization problem.

2.1 Margin

Given a linear decision boundary defined by $w^T x + b = 0$. The functional margin for a point (x^i, y^i) is defined as

$$y^i(w^T x^i + b)$$

For a fixed w and b, the larger the functional value, the more confident we have about the prediction. However, the functional margin can be arbitrarily changed without changing the boundary at all. So we use geometric margin instead. **Geometric Margin** The distance between an example and the decision boundary. For training set $S = \{x^i, y^i\} : i = 1, ...N$ and boundary $w^T + b = 0$, compute the geometric margin of all points:

$$\gamma^i = \frac{y^i(W \cdot X^i + b}{||W||}, i = 1, ..., N$$

Note: $\gamma^i > 0$ if point i is correctly classified We want to see if the smallest γ^i is large.

$$\gamma = \min_{i=1...n} \gamma^i$$

2.2 Optimization Problem

Maximizing the geometric margin is equivalent to minimizing the magnitude of w subject to maintaining a functional margin of at least 1.

$$\min_{w,b} \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } : y^i \left(\mathbf{w} \cdot \mathbf{x}^i + b\right) \geq 1, i = 1, \cdots, N$$

Results in a quadratic optimization problem with linear inquality constraint. There are several algorithms for solving for QP. We can regard them as black box. The solution can be written in forms of

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y^i x^i, \quad s.t. \sum_{i=1}^{N} \alpha_i y^i = 0$$

The above equation provide the form for w, the value of b can be computed with some additional steps

- w is a linear combination fo all training exampls
- many points have zero α's, which are the data points that have larger geometric margin
- points that have non-zero α's are called **support vector**, which are the data points that have smallest geometric margin

2.3 Dealing with none linearly separable data

If data are not linearly separable or data have noise. It becomes difficult to use SVM. We have two ways to deal with these issues.

2.4 Soft Margin

Allow functional margin to be less than 1, or in some cases less than 0 . Adding the software margin to our equation, we have

$$\min_{w,b} \|\mathbf{w}\|^2 + c\sum_{i=1}^{N} \xi_i^k$$

$$\text{subject to } : y^i \left(\mathbf{w} \cdot \mathbf{x}^i + b\right) \geq 1 - \xi_i, i = 1, \cdots, N$$

$$\xi_i \geq 0, i = 1, \cdots, N$$

With solution of

$$w = \sum_{i=1}^{N} \alpha_i y^i x^i, \quad s.t \sum_{i=1}^{N} \alpha_i y^i = 0 \text{ and } 0 \leq a_i \leq c$$

- ξ can be viewed as errors
- Tradeoff between maxmizing decision boundary margin and minimizing error
- Parameter c controls this tradeoff, c also puts a box constraints on α and limits the influence of individual support vector
- C is set by the algorithm implementer, and can be derived using cross-validation

2.5 Mapping input space to higher dimension

When dataset is too hard to seperate linearly using soft margin. We can map the input space to higher dimension such that the data points become linearly seperatable.

2.6 Kernel Function

Kernel function is a function that maps input space to higher dimension. It can also be viewed measuring similarity. As a result, the decision boundary will be non-linear in the original input space.

- There are many kernel functions, the choice can be derived by cross-validation

Strengths

- solution is globally optimal
- Scales well with higher dimensional data
- Can handle non-traditional data like strings, trees

Weakness

- Need to choose a good kernel
- Can be computational expensive for large dataset

3 Decision Tree

Use a tree structure for solving classification problems. Its strengths are

1. Similar to human decision, high interpretability
2. Deals with discrete and continuous features without the need for transformation unlike perceptron and logistic regression
3. Highly flexible, can represent more complex decision boundaries by increasing nodes and depth

The learning objective using decision tree is to find a decision tree h that achieves minimum error on training data.

3.1 Algorithm

A top-down, greedy search approach

1. Choose the best test to be the root of tree
2. Create a descendant node for each test outcomes
3. Examples in training set S are sent to the appropriate descendent node based on the test outcome
4. Recursively apply at each descendent node using the subset of training samples
5. If all samples belong to the same class, turn it into a leaf node

Choosing the best test, we aim to maximize the information gain. In other words, minimize entropy.

3.2 Entropy

Entropy is a **measure of uncertainty**. If the probability is 1.0, there is no entropy. However, if the probability of an outcome over another is 0.5, the entropy is maximized.

Let y be a categorical random variable that can take k different values: $v_1, v_2, ..., v_k$ and $p_i = P(y = v_i)$ for $i = 1, .., k$. The entropy of y, denoted as $H(y)$ is defined as:

$$H(y) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

3.3 Building decision tree

We need to choose the split that maximizes **benefit of split** which effecitvely measures the mutual information between the features x and class label y. The root is then selected based on information gain.

$$\text{Benefit of split } = U(S) - \sum_{i}^{m} p_i U(S_i)$$

3.4 Dealing with multi-nomial features and continuous features

Multi-nominal: If a feature has more than two possible values.

- can be problemic because there is a bias to prefer multinominal features to binary features.

Continuous features

- Compute a threshold that maximizes information gain, essentially convert it to a binary feature
- Both continuous features and discrete features can be used to formulate a decision tree

3.5 Over-fitting

Due to being highly flexible, the decision tree is prone to over-fitting. Two interventions can combat that

1. Early stop
    - stop growing the tree when data split does not offer large benefits
2. Post-pruning
    - Separate training data into a training set and validating set
    - Compute the impact on validation set when pruning each possible node
    - Prune the node that improves the validation set performance in a greedy fashion

3.6 Regression Tree

Using decision tree to apply for regression problems. Prediction is computed as the average of the target values of all examples in the leaf node. Uncertainty is measured by the sum of squared errors within the node.

4 Questions

1. The mechanism of Kernel function in SVM in mapping to higher dimension?
2. The concept of information gain in decision tree?
3. In choosing between linear models and SVM? Can overfitting be an issue in SVM?

5 Machine Learning Handson

Investigate a dateset of cancer microarray, and use machine learning model to to predict the outcome

5.1 Make machine learning research reproducible, make it pubic

- All optimal tuning parameters chosen for each technique evaluated
- The pseudocode for the data partitions
- The number of replicates performed to obtain the average test errors
- The seed used as the entry point into the random number generator during replication process

6 Feature engineering

6.1 Feature subset selection

Removing features that are not relevant or are redundant, very helpful for high dimensional data. There are three type of feature selection algorithms

6.2 The filters which extract features from the data without any learning involved

Gene ranking as a popular statistical method, which ranks gene in the dataset by their significance

- Unconditional Mixture Modelling (univariate): assume two different states of gene on and off, and checks whether the underlying binary state of gene affects the classification using mixture overlap probability

- Information Gain Ranking (univariate): approximates the conditional distribute P(C|F), where C is the class label and F is the feature vector. Information gain is used as a surrogate for the conditional distribution
- Markov Blanket Filtering (univariete): finds the features that are independent of the class label so that removing them will not affect the accuracy

6.3 Dealing with missing values

Three ways

1. Remove the column with missing values, may be used if column contains mostly missing values
2. loss of information
3. Imputation: replace the missing value with some number. Scikit-learn's imputation library replace values with mean by default.
4. a better option most of time

6.4 Dealing with Categorical data

- One-Hot Encoding: create new binary columns for each category

7 Visualization of High Dimensional Data

7.1 TSNE

- TSNE: converts similarities between data point to join probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data

8 Experiment Design for ML algorithm Evaluation

Sparse: when a feature have most its entries as zeros.

- sparse maxtrix: a matrix contains mostly zero values
- dense matrix: a matrix contains mostly non-zero values

8.1 Why high sparsity is desired in many ML applications?

1. Many real datasets such as texts and Microarray data are represented as very high dimensional vectors
2. Most features in high dimensional vectors are usually non-informative or noisy and may serious affect the generalization performance
3. A sparse classifier can lead to a simplified decision rule for faster prediction in large-scale problems

8.2 Performance Matrice

- accuracy
- AUC: area under ROC curve
- squared loss

8.3 Models that perform well across low-dimension to high-dimension data

- Random Forest, Neural nets, Boosted Tree, and SVMS

9 XGBoost

XGBoosst is the leading model for working with standard tabular data (eg. in Pandas Dataframe). It requires more knowledge and model tuning. It is an implementation of the gradient boosted decision trees algorithm.

- start with a baseline prediction, create cycles that repeatedly builds new models and combines them into an ensemble model
- to make a prediction, we add the predictions from all previous models
- n estimator is key parameter to tune, too small leads to underfitting, and too large leads to overfitting
- early stopping rounds is another parameter can stop the algorithm automatically when model stops improving
- learning rate allows early predictions to have smaller weight, and later ones have a larger weight. So we can use a larger n estimator value with learning rate

- n jobs is a parameter can be set to number of cores to take advantage of parallism

10 Pipelines

A pipeline bundles preprocessing and modeling steps so you can use the whole bundle as if it were a single step. It involves defining the steps of applying transformors to the data, then train the models. Benefits include

1. cleaner code
2. fewer bugs
3. easier to productionize
4. more options for model testing

11 Cross Validation

Provide a more accurate measure of model quality by fold the data into partitions.

- lower score means better model quality

12 Data leakage

Leakage causes a model to look accurate until making predictions with the model. Any variable that updates after target values is realized can cause data leakage, so they should be excluded from the feature set.

- Data leakage can cause major problem in ML production, need to be careful

13 General Visualization

13.1 Partial Dependence Plots

Show how each variable or predictor affects the model's predictions after the model is fitted. Improve the interpretability.

14 Debugging and Error Analysis

Error analysis is crucial when applying machine learning to real world problems. [`http://cs229.stanford.edu/materials/ML-advice.pdf`]

14.1 Debugging learning algorithms

14.1.1 high variance or overfitting: if error is low on training but high on testing. Overlearning the training data but no generalizing well

- increase training samples
- a smaller set of features

14.1.2 high bias: if error is high on training regardless of training samples. Not enough features to learn the problem

- a larger set of features
- design better features

14.1.3 algorithm not converging

- run gradient descent for more iterations
- use Newton's method

14.1.4 Not optimizing the right objective function

- parameter tuning current algorithm
- try another algorithm

14.2 Error analysis vs. ablative analysis

14.2.1 Error analysis tries to explain the difference between current performance and perfect performance.

14.2.2 Ablative analysis tries to explain the difference between some baseline performance and current performance. Eg. consider a image recognition algorithm with baseline performance of 94%, and best performance of 99%.

14.3 Two approaches to ML problems

1. careful design: try to design the right features, dataset and algorithm architecture.
    - pro: maybe more scalable
    - con:Be careful with premature optimization
2. build and fix: implement something quick, then run error analyses to fix its errors.
    - pro: faster to market

15 misc

15.1 treatment effect

The difference between treated and untreated group

15.2 parameteric

Assume data is drawn from normal distribution

15.3 non-parameteric

Does not assume data to have normal distribution

15.4 unconfoundedness

An assumption that confounding variables are measured in the dataset

16 Foundations

16.1 Probability Theory

16.1.1 Bayesian Probability

An intepretation of the concept of probability. Instead of frequency or propensity of some phenomenon, probability is intepreted as reasonable expectation representing a state of knowledge or as quantification of a personal belief.

- Bayes' therom to convert prior probability to posterior probability based on evidences provided by the observed data
- The effect of observed data $D = t_1, ..., t_n$ is expressed through conditional probability $p(D|w)$

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

$p(D|w)$ is learned from observed data, is called the **likelihood function**. Given the definition of likelihood. We state Bayes' therom as

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

- negative log of the likelihood function is called an **error function**. Because error function is monotonically decreasing, maximizing the likelihhood is equivalent to minimizing the error
- bootstrap: suppose orginal data set consists of N data points at random from X. We create a new dataset X_b by drawing N points at random from X, with replacement, so some points in X may be replicated in X_B, where other points in X may be absent from X_B. This process is repeated L times to generate L data sets each of size N and each obtained by sampling from the original data set X
- One advantage of the Bayesian viewpoint is that inclusion of prior knowledge arises naturally. Suppose, that a fair-looking coin is tossed three times and lands head each time. A classical maximum likelihood estimate of the probability of landing heads would give 1. By contrast, a Bayesian approach with any reasonable prior will lead to much less extreme conclusion

16.1.2 The Gaussian distribution

For a single real-valued variable x, the Gaussian distribution is defined by

$$\mathcal{N}\left(x|\mu,\sigma^2\right) = \frac{1}{\left(2\pi\sigma^2\right)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

Two key parameters are σ^2, the standard deviation and μ, the mean. The reciprocal of variance is called precision, written as $\beta = 1/\sigma^2$

16.1.3 Curve fitting

16.2 Model Selection

16.3 The Curse of Dimensionality

16.4 Decision Theory

16.5 Information Theory

17 Probability Distributions