Home Assignment 4

Yu Wang(ndp689)

December 21, 2021

Contents

1 Support Vector Machines

1.1 Data understanding and preprocessing

1) Report the class frequencies:

   For the class labeled 1, the class frequency is 0.5467
   For the class labeled -1, the class frequency is 0.4533

2) Number of training and test examples:

   Number of training examples: 150
   Number of test examples: 164

3) Code snippets for the normalization:

```
# compute the mean and standard deviation of the training features
mean_of_training = np.mean(X_train, axis=0)
standard_of_training = np.std(X_train, axis=0)
# to normalize data
norm_train = (X_train - mean_of_training) / standard_of_training
norm_test = (X_test - mean_of_training) / standard_of_training
```

Figure 1: Normalization

4) Mean and variance of features in the test data

```
Mean of normalized features in the test data:
[ 0.09  0.17 -0.06 -0.08 -0.04 -0.11 -0.1  -0.21  0.27  0.08  0.01  0.06
  0.01  0.    0.13  0.02  0.13  0.13  0.03  0.1   0.48  0.11  0.05 -0.12
  0.11  0.02 -0.1  -0.13 -0.18  0.01 -0.03  0.    0.2  -0.01 -0.08  0.17
  0.3   0.18  0.05 -0.02  0.08  0.22  0.04 -0.12 -0.03  0.1   0.12  0.1
 -0.07 -0.05 -0.13  0.04 -0.    0.01  0.23 -0.04  0.14  0.14  0.04 -0.01
 -0.06]
Variance of normalized features in the test data:
[ 1.93  7.28  0.79  0.74  0.86  0.98  1.07  2.88  2.97  1.48  1.09  1.14
  1.12  1.24  1.27  1.01  1.13  3.89  5.71  5.   54.28  1.44  1.02  0.97
  0.85  1.16  0.59  0.8   0.4   1.22  1.03  1.    1.03  1.07  0.82  4.91
 11.01  0.97  0.81  0.89  2.44  2.21  1.51  0.89  1.32  0.82  1.2   2.23
  1.22  0.93  1.19  1.31  1.39  0.86  1.94  1.03  1.07  1.21  1.74  1.87
  1.01]
```

Figure 2: Mean and variance of features

1.2 Model selection using grid-search

1) Description of software used

   I use the 'svm.SVC' in sklearn to create the SVM model, and let the kernel be Gaussian kernel. For the two hyperparameters γ and C, I choose to vary their values on the scales (0.0001, 0.001, 0.01, 0.1, 1, 10, 100) and (0.01, 0.1, 1, 10, 100, 1000, 10000) respectively.

2) A short description of how you proceeded (e.g., did the cross-validation)

   I use 'GridSearchCV' in sklean to do the cross-validation and to pick the best hyperparameters, where let the 'cv' be 5 to do the 5-fold cross validation. And the details are as follows:

```
# 5-fold cross-validation using grid-search
grid = GridSearchCV(svm.SVC(), param_grid = params, cv=5, scoring='accuracy')
grid.fit(norm_train, y_train)

best_C = grid.best_params_["C"]
best_gamma = grid.best_params_["gamma"]
print('the best C:',  str(best_C))
print('the best gamma:',  str(best_gamma))
```

Figure 3: Doing the 5-fold cross validation

To predict the classes of the test dataset, I train an SVM model using the complete training dataset, Gaussian kernel, and the best C and γ determined above. And then, I use the `accuracy_score` function to get the accuracy of the predictions. And the details are as follows:

```python
# train an SVM with the best hyperparameters using the complete training dataset
model = svm.SVC(kernel = 'rbf', C = best_C, gamma = best_gamma)
model.fit(norm_train, y_train)

# training error
training_predictions = model.predict(norm_train)
training_accurracy = accuracy_score(y_train, training_predictions)
print('training error:', 1 - training_accurracy)

# test error
test_predictions = model.predict(norm_test)
test_accurracy = accuracy_score(y_test, test_predictions)
print('test error:',1 - test_accurracy)
```

Figure 4: Computing the errors

3) Training and test errors as well as the best hyperparameter configuration

   As the Figure 3 shows, using the grid-search, we can determine appropriate SVM hyperparameters γ and C: C = 1 and $\gamma = 0.01$.

   The training error from the trained SVM model is 0.0467 and the test error is 0.2073.

1.3 Inspecting the kernel expansion

1) Answer and rigorous argumentation

   The number of bounded and free support vectors will drastically change if C is drastically increased and decreased.

2) Results of empirical validation including description of how these results were computed

   Results: Let $C \in \{0.001, 10000\}$. When C = 0.001, the number of bounded SV is 136 and the number of free SV is 0. And when C = 10000, the number of bounded SV is 0 and the number of free SV is 72.

   Description: First, I use 'svm.SVC' to train a model using the complete training dataset, Gaussian kernel, and the γ (using two different values of $C \in \{0.001, 10000\}$). And then I use `dual_coef_` to get the $\alpha_i y_i$ and use np.abs($\alpha_i y_i$) to get the values of all α_i. For bounded SV, the value of α = C and for free SC, the value of $\alpha \in (0, C)$. Then, we can get the number of bounded SV and free SV.

2 The airline question

1. Let X be the number of people who will not show up, and $X_i \in X(X_i \in \{0,1\})$ be the independent random variable. We define X $\sim$ B(100,0.05).

   Then, in this question, the value of X should meet X = 0. So, we have:

   $$P(X=0) = 0.05^0 \times (1-0.05)^{100} \approx 0.59\%$$

2. (a) In the first approach, let the first event be E_1 and the second event be E_2.

   In E_1, we let X be the number of people who show up, and $X_i \in X(X_i \in \{0,1\})$ be the independent random variable. So, we need to compute $P(E_1) = P(\sum_{i=1}^{10000} X_i = 9500)$, and I will use Hoeffding's inequality to compute its bound:

   $$\begin{aligned} P(E_1) &\leq P(\frac{1}{10000}\sum_{i=1}^{10000} X_i \geq 0.95) \\ &= P(\frac{1}{10000}\sum_{i=1}^{10000} X_i - p \geq 0.95 - p) \\ &\leq e^{-2\times 10000 \times (0.95-p)^2} \end{aligned}$$

   In E_2, let X be the number of people who show up, and $X_i \in X(X_i \in \{0,1\})$ be the independent random variable. We define X $\sim$ B(100,p). So, we need to compute $P(E_2) = P(X = 100) = p^{100}$. Since E_1 and E_2 are independent, the the probability that they happen simultaneously is $P(E_1)P(E_2)$:

   $$P(E_1)P(E_2) \leq e^{-2\times 10000 \times (0.95-p)^2} \times p^{100} \quad (1)$$

   Using equation (1) we can visualize the problem:

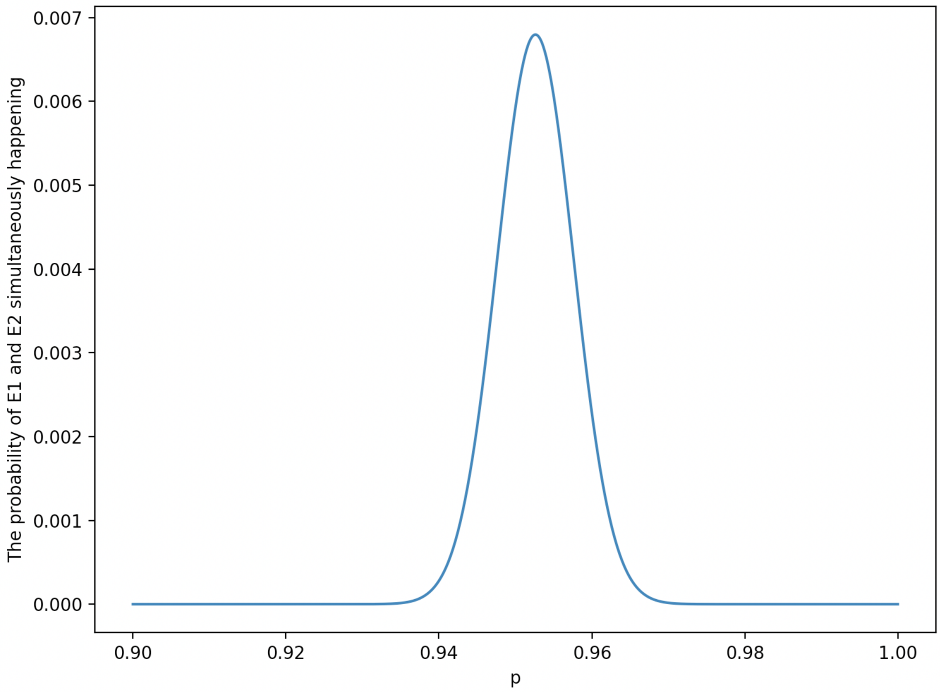

Figure 5: The probability of E1 and E2 simultaneously happening

From Figure 1, we know that when $p \approx 0.9527$ we can get the the worst case, and the bound of probability that the two events happen simultaneously is about 0.006797

(b) We let $S \cup S' = 10100$ be the 10100 sampled passengers, and we split it into $S = 10000$ (the 10000 passengers in the collected sample) and $S' = 100$ (the 100 passengers booked for the 99-seats flight).

And we let A be the event of a sample of 10000 with 95% show ups, and B be the event of a 99-seats flight with all 100 passengers showing up. In this question, we need to compute the bound of P(AB). We have:

$$
\begin{aligned}
P(AB) &= \sum_{S \cup S'} P(S \cup S') P(AB|S \cup S') \\
&\leq \sup_{S \cup S'} P_{split}(AB|S \cup S') \\
&= \left(\frac{10000 \times 95\% + 100}{10100}\right)^{100} \approx 0.00623
\end{aligned}
$$

* The best splitting is 9600 passengers show up in sample 10100, and for the 9600 passengers, 9500 passengers are in event A and 100 passengers are in event B.

3 The growth function

1. The definition of the growth function of $\mathcal{H}$ is:
$$m_{\mathcal{H}}(n) = \max_{x_1,...,x_n} |\mathcal{H}(x_1,...,x_n)|,$$
$$where: \ \mathcal{H}(x_1,...,x_n) = \{(h(x_1),...,h(x_n)) : h \in \mathcal{H}\}$$
Then, the maximal number of dichotomies generated by $\mathcal{H}$ on $(x_1,...,x_n)$ is 2^n. So, $m_{\mathcal{H}}(n) \le 2^n$;

   Besides, for each $h \in \mathcal{H}$, it can classify $(x_1,...,x_n)$ to one result. Because $|\mathcal{H}| = M$, we have $m_{\mathcal{H}}(n) \le M$;

   So, $m_{\mathcal{H}}(n) \le \min\{2^n, M\}$.

2. From Point 1, we know that $m_{\mathcal{H}}(n) \le \min\{2^n, M\}$.

   If $2^n < M$, $n < 1$. Because in the growth function of H, the value of n needs to be larger than or equal to 1, $2^n < M$ is impossible.

   If $2^n > M$, $m_{\mathcal{H}}(n) \le M = 2$. Because $\mathcal{H}$ have two different hypothesis, they can produce two labels, which means $m_{\mathcal{H}}(n) \ge 2$. Then, we have $m_{\mathcal{H}}(n) = 2$.

3. Let $d_{VC}(\mathcal{H}) = k$.

   If $k < n$, then we can use the Sauer's Lemma:
$$m_{\mathcal{H}}(n) = \sum_{i=0}^{d_{VC}(\mathcal{H})=k} \binom{n}{i} \le n^k + 1$$
Then, $m_{\mathcal{H}}(2n) \le (2n)^k + 1 \le n^{2k} + 1 = m_{\mathcal{H}}(n^2)$.

   If $k \ge n$, then:
$$\begin{aligned} m_{\mathcal{H}}(n) &= \sum_{i=0}^{d_{VC}(\mathcal{H})=k} \binom{n}{i} \\ &= \sum_{i=0}^{n} \binom{n}{i} = 2^n \end{aligned}$$
Then, $m_{\mathcal{H}}(2n) = 2^{2n} = m_{\mathcal{H}}(n^2)$.

   So, we have proved that $m_{\mathcal{H}}(2n) \le m_{\mathcal{H}}(n^2)$