Validation and Interpretation

Cross Validation and Bootstrap
Recap

• We are going to set out to answer some of the big remaining questions around model performance
  – Monday/Wednesday:
    • How do we obtain reliable performance estimates?
      – Motivation and Methods commonly used to assess model performance
  – Friday:
    • How do we measure the performance of a model?
      – Discussion of various performance metrics and their uses
Recap: A Theoretical Foundation

Bias Variance Tradeoff

- Low Bias, Low Variance
- High Bias, High Variance
Recap: Generalization

Low variance, high bias method

Underfitting:
• Model is too “simple” to represent all the relevant class characteristics
  – High bias and low variance
  – High training error and high test error

Low bias, high variance method

Overfitting:
• Model is too “complex” and fits irrelevant characteristics (noise) in the data
  – Low bias and high variance
  – Low training error and high test error

Underfitting:
• Model is too “simple” to represent all the relevant class characteristics
  – High bias and low variance
  – High training error and high test error

Overfitting:
• Model is too “complex” and fits irrelevant characteristics (noise) in the data
  – Low bias and high variance
  – Low training error and high test error
Recap: Estimating Model Performance

How do we estimate performance measures?
Can I just compute distance of each point to the true value?

No!
• These data points were used to create the line of best fit.
• Called resubstitution error.
• Not a good indicator of the performance on future data.

Simple solution
• Spit the available data into training and testing sets.
• This allows us to test “unseen” data without having to collect new data samples.
Illustrating Why Training Data Matters
A Typical Model Setup

1. **Data**
2. **Training Set**
3. **Test Set**
4. **Learning Algorithm**
   - **Learn Model**
   - **Apply Model**
5. **Model**

- **Induction** from Training Set to Learn Model
- **Deduction** from Test Set to Apply Model
Recap: Breaking Data into Test Sets

• Holdout
  – Single holdout
  – Stratified holdout
  – Repeated holdout

• Cross validation
  – Leave-one-out validation
  – $k$-fold validation
  – Stratified $k$-fold validation

• Bootstrap
Recap: Holdout Estimation

Key Idea:

Reserve a certain amount of data for testing and use the remainder for training.

\[ \text{Train Data} \quad 70\% \quad \text{Test Data} \quad 30\% \]
Recap: Simple / Stratified holdout

• In Simple holdout data is randomly selected from the original data and broken (often at a specific percentage) into a training and testing dataset.

• Often it is suggested we first shuffle the data
  – Helps to prevent against any potential collection bias due to time or related instances

• Generate holdout using *stratified sampling*.
  – New subsets of instances have approximately equal proportions of classes

• Ensures classes are equally represented in samples.
Recap: Repeated Holdout

• Repeated holdout, improves the reliability of the holdout estimate by repeating the process with different subsamples.
  – Also known as “random subsampling.”

• For a set number of iterations, the same proportion of data is randomly selected for training utilizing either simple or stratified holdout method.
  – Error rates on different iterations averaged to yield overall error.
Breaking Data into Test Sets

• Holdout
  – Single holdout
  – Stratified holdout
  – Repeated holdout

• Cross validation
  – Leave-one-out validation
  – $k$-fold validation
  – Stratified $k$-fold validation

• Bootstrap (Extra)
Cross – Validation

Key Idea:

**Building on repeated holdout, enforce splits that place each data point in the test data only once.**
Leave-one-Out Cross-Validation (LOOCV)

- The simplest way to ensure test sets do not overlap is to evaluate each data point as its own test set.
- Number of iterations equals the number of instances. Classifier trained $n$ times, where $n$ is number of training instances.
- Typically used when number of instances is small, as it provides the maximal amount of training data.
Cross Validation Error

\[ E_{cv} = \frac{1}{3}(e_1 + e_2 + e_3) \]
Leave-one-Out Cross-Validation: Challenges

• **Bias-Variance Tradeoff:**
  – Less Bias (almost all the data is used each time)
  – High variance of the true error rate estimator will be large

• **Selecting only one instance can result in poor performance estimates.**
  – Consider:
    • Random dataset split equally into two classes
    • Best model predicts majority class
    • 50% accuracy on unseen data
    • 100% error using leave-one-out cross-validation!

• **Additionally: is computationally expensive.**
  – Can be a concern if the estimation technique is not efficient
Leave More Than One Out

- To provide more robust estimation of error we would like to increase the number of points in each test dataset
  - Whereas: Leave one out: \( N \) training sessions on \( N - 1 \) points each

- We can expand into \( k \) (usually 5 or 10) equal disjoint subsets or *folds*.
  - e.g., 10 folds, 1 fold (10% of data) used for testing, 9 folds (90% of data) used for training
  - Testing fold rotates each iteration with \( \frac{N}{k} \) training sessions on \( N - K \) points each

- Average results over all 10-folds for error estimate.
  - To further reduce variance in estimates, may repeat.
Random Cross-Validation

- Data randomly divided into $k$ equal folds.
  - Advantage is the train/test division does not depend on the number of iterations.
  - Disadvantage is that some instances may not be used and others may be used more than once.
Stratified Cross-Validation

- More commonly: Data divided into $k$ equal folds, each with a representative number of class values.
  - Proportion of instances for each class is representative.
  - Each instance belongs to only one fold.
- Advantage is that each fold is representative.
Properties of Cross-Validation

• Uses sampling \textit{without} replacement.
  – The same instance, once selected, cannot be selected again for a particular training/testing set

• Since each training set is only \((K - 1)/K\) as big as the original training set, the estimates of prediction error will typically be biased upward.
  – This bias is minimized when \(K = n\) (LOOCV), but this estimate has \textbf{high variance} (remember the tradeoff!)

• In general Cross-Validation tends to have high variance
Validation Logistics: Transformations

Who remembers this example?

Why does it present a problem for distance comparisons?

<table>
<thead>
<tr>
<th>Patient</th>
<th>Total Eye Blinks</th>
<th>Heart Attacks</th>
<th>Tumor Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25000</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>40000</td>
<td>27</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>55000</td>
<td>32</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>27000</td>
<td>25</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>53000</td>
<td>30</td>
<td>5</td>
</tr>
</tbody>
</table>
Validation Logistics: Transformations

*Why does it present a problem for distance comparisons?*

- Scales were vastly different,
  - The eye blinks dominate crude calculations of Euclidian distance

**Solution:**

Transform the data, normalizing features onto the same scale

<table>
<thead>
<tr>
<th>Patient</th>
<th>Total Eye Blinks</th>
<th>Heart Attacks</th>
<th>Tumor Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25000</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>40000</td>
<td>27</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>55000</td>
<td>32</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>27000</td>
<td>25</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>53000</td>
<td>30</td>
<td>5</td>
</tr>
</tbody>
</table>
Validation Logistics: Transformations

Min-Max Normalization

- Transform the data from measured units to a new interval from \( new\_minF \) to \( new\_max \)
  - \( V \) is the current value of feature \( F \).

\[
v' = \frac{v - min_F}{max_F - min_F} (new\_max_F - new\_min_F) + new\_min_F
\]

Z-score Normalization

- Transform the data by converting the values to a common scale with an average of zero and a standard deviation of one.
  - Value, \( v \), of \( A \) is normalized to \( v' \) by computing:

\[
v' = \frac{v - \bar{v}}{\sigma_f}
\]

What is the challenge with test/train splits here?
Validation Logistics: Transformations

Min-Max Normalization

- Transform the data from measured units to a new interval from new_minF to new_max
  - V is the current value of feature F.

\[ v' = \frac{v - \text{min}_F}{\text{max}_F - \text{min}_F} (\text{new_max}_F - \text{new_min}_F) + \text{new_min}_F \]

Z-score Normalization

- Transform the data by converting the values to a common scale with an average of zero and a standard deviation of one.
  - Value, v, of A is normalized to v' by computing:

\[ v' = \frac{v - \bar{v}}{\sigma_f} \]

What is the challenge with test/train splits here?

We are normalize the column using data we are going to hold out!

This is an example of “data snooping”, where we try and predict on unseen examples but we give the classifier information anyway (i.e. the true scale of features)
Validation Logistics: Transformations

Solution

Compute the transformation bounds on the training data only

Then only transform the test data before evaluating, using the known bounds transformation

MinMaxScaler.fit_transform()

MinMaxScaler.transform()
Validation Logistics: Class Imbalance

- Choosing the number of folds is also a highly subjective practice
  - Although commonly 5 and 10 are used, some thought should be placed on which is more appropriate for your data

- Imagine for a minute you have highly imbalanced data
  - Many instances of class 0, and few of class 1

- By choosing 10 fold it is much less likely that a minority class element appears in a test set (as the sets are smaller)
  - Thus it becomes a balance between percentage of test data, and fold-count
  - Even for stratified sampling the number of minority instances in the test set may be very low, increasing your overall variance
Breaking Data into Test Sets

• Holdout
  – Single holdout
  – Stratified holdout
  – Repeated holdout

• Cross validation
  – Leave-one-out validation
  – $k$-fold validation
  – Stratified $k$-fold validation

• Bootstrap (Extra)
**Bonus: Bootstrap Sampling**

**Key Idea:**

*In an effort to reduce the variance of CV, we can sample with replacement to better estimate our population’s true data distribution*
The Bootstrap

• The bootstrap approach allows us to mimic the process of obtaining new data sets

• Rather than repeatedly obtaining independent data sets from the population
  – Sample dataset of $n$ instances $n$ times with replacement, forming new dataset of $n$ instances
  • As a result some observations may appear more than once in a given bootstrap data set and some not at all
The Bootstrap Technicalities

• An instance has probability of $1 - 1/n$ of not being selected for training.
• Given $n$ selections, probability of being in test data is:

$$\left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0.368$$

• Training data will contain about 63.2% of the instances.
Estimating Error using Bootstrap

- The bootstrap analogue to cross validation estimates of generalization error is called *out-of-bootstrap* estimate – i.e. the test cases are those that were left out of the bootstrap resampled training set.
Properties of the Bootstrap

• Error estimate on test data will be pessimistic
  – Training is performed on only ~63% of the instances.
  – \[ \text{error} = 0.632 \times e_{\text{testing}} + 0.368 \times e_{\text{training}} \]

• For small sample sizes, bootstrap has smaller variability than cross-validation estimates

• For large sample sizes, bootstrap and CV estimates will generally be close.
  – As sample size approaches infinity, will approach unity.
Consider random dataset with 50% class distribution.

- Model that memorizes training data will achieve 0% resubstitution error on training data and 50% error on testing data.

- Bootstrap estimate:
  
  \[
  error = 0.632 \times 0.5 + 0.368 \times 0 = 31.6\%
  \]
Next Class

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>Positive</td>
</tr>
<tr>
<td></td>
<td>TP</td>
</tr>
<tr>
<td>Negative</td>
<td>FN</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
</tr>
<tr>
<td></td>
<td>FP</td>
</tr>
<tr>
<td></td>
<td>TN</td>
</tr>
</tbody>
</table>

False Positive Rate vs. True Positive Rate

TP: True Positive
FN: False Negative
FP: False Positive
TN: True Negative