Experimental Setup, Multi-class vs. Multi-label classification, and Evaluation

CMSC 678
UMBC
Central Question: How Well Are We Doing?

Classification

Regression

Clustering

*the task*: what kind of problem are you solving?

- Precision, Recall, F1
- Accuracy
- Log-loss
- ROC-AUC
- ...

- (Root) Mean Square Error
- Mean Absolute Error
- ...

- Mutual Information
- V-score
- ...

- Precision, Recall, F1
- Accuracy
- Log-loss
- ROC-AUC
- ...
Central Question: How Well Are We Doing?

The task: what kind of problem are you solving?

Classification
- Precision
- Recall
- F1
- Accuracy
- Log-loss
- ROC-AUC
- ...

Regression
- (Root) Mean Square Error
- Mean Absolute Error
- ...

Clustering
- Mutual Information
- V-score
- ...

This does not have to be the same thing as the loss function you optimize.
Outline

Experimental Design: Rule 1

Multi-class vs. Multi-label classification

Evaluation

Regression Metrics
Classification Metrics
Experimenting with Machine Learning Models

All your data

Training Data  Dev Data  Test Data
Rule #1

DEVELOP ON DEV DATA

DON'T ITERATE ON YOUR TEST DATA
Experimenting with Machine Learning Models

What is “correct?”

What is working “well?”

- Training Data
- Dev Data
- Test Data

Learn model parameters from training set
Experimenting with Machine Learning Models

What is “correct?”

What is working “well?”

Training Data

Dev Data

Test Data

Evaluate the learned model on dev with that hyperparameter setting

Learn model parameters from training set

set hyper-parameters
Experimenting with Machine Learning Models

What is “correct?”

What is working “well?”

Evaluate the learned model on dev with that hyperparameter setting

Learn model parameters from training set

perform final evaluation on test, using the hyperparameters that optimized dev performance and retraining the model
Experimenting with Machine Learning Models

What is “correct?”
What is working “well?”

Rule 1: DO NOT ITERATE ON THE TEST DATA
On-board Exercise

Produce dev and test tables for a linear regression model with learned weights and set/fixed (non-learned) bias
Outline

Experimental Design: Rule 1

Multi-class vs. Multi-label classification

Evaluation

Regression Metrics

Classification Metrics
Multi-class Classification

Given input $x$, predict discrete label $y$

Multi-label Classification
Given input $x$, predict discrete label $y$.

If $y \in \{0,1\}$ (or $y \in \{\text{True, False}\}$), then a binary classification task.
Multi-class Classification

Given input $x$, predict discrete label $y$

If $y \in \{0,1\}$ (or $y \in \{\text{True, False}\}$), then a binary classification task

If $y \in \{0,1, \ldots, K - 1\}$ (for finite $K$), then a multi-class classification task

Q: What are some examples of multi-class classification?
Multi-class Classification

Given input $x$, predict discrete label $y$

- If $y \in \{0,1\}$ (or $y \in \{\text{True, False}\}$), then a binary classification task
- If $y \in \{0,1,\ldots,K-1\}$ (for finite $K$), then a multi-class classification task

Q: What are some examples of multi-class classification?

A: Many possibilities. See A2, Q{1,2,4-7}

Multi-label Classification
## Multi-class Classification

Given input $x$, predict discrete label $y$

<table>
<thead>
<tr>
<th>Single output</th>
<th>If $y \in {0,1}$ (or $y \in {\text{True, False}}$), then a binary classification task</th>
<th>If $y \in {0,1, \ldots, K - 1}$ (for finite $K$), then a multi-class classification task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-output</td>
<td>If multiple $y_l$ are predicted, then a multi-label classification task</td>
<td></td>
</tr>
</tbody>
</table>

## Multi-label Classification
Multi-class Classification

Given input $x$, predict discrete label $y$:

- **Single output**
  - If $y \in \{0,1\}$ (or $y \in \{\text{True}, \text{False}\}$), then a binary classification task.
  - If $y \in \{0,1,\ldots,K-1\}$ (for finite $K$), then a multi-class classification task.

- **Multi-output**
  - If multiple $y_l$ are predicted, then a multi-label classification task.

Given input $x$, predict multiple discrete labels $y = (y_1, \ldots, y_L)$:

Multi-label Classification
**Multi-class Classification**

Given input $x$, predict discrete label $y$

<table>
<thead>
<tr>
<th>Single output</th>
<th>If $y \in {0,1}$ (or $y \in {\text{True, False}}$), then a binary classification task</th>
<th>If $y \in {0,1, \ldots, K-1}$ (for finite $K$), then a multi-class classification task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-output</td>
<td>If multiple $y_l$ are predicted, then a multi-label classification task</td>
<td>Each $y_l$ could be binary or multi-class</td>
</tr>
</tbody>
</table>

Given input $x$, predict multiple discrete labels $y = (y_1, \ldots, y_L)$

**Multi-label Classification**
Multi-Label Classification...

Will not be a primary focus of this course

Many of the single output classification methods apply to multi-label classification

Predicting “in the wild” can be trickier

Evaluation can be trickier
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

(there can be others)
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

Loss function may (or may not) need to be extended & the model structure may need to change (big or small)

(there can be others)
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

(there can be others)

Loss function may (or may not) need to be extended & the model structure may need to change (big or small)

Common change:

instead of a single weight vector $w$, keep a weight vector $w^{(c)}$ for each class $c$

Compute class specific scores, e.g.,

$$y^{(c)}_i = (w^{(c)})^T x + b^{(c)}$$
Multi-class Option 1: Linear Regression/Perceptron

\[ y = \mathbf{w}^T \mathbf{x} + b \]

output:
if \( y > 0 \): class 1
else: class 2
Multi-class Option 1: Linear Regression/Perceptron: A Per-Class View

\[ y = \mathbf{w}^T \mathbf{x} + b \]

output:
if \( y > 0 \): class 1
else: class 2

data version is a special case

\[ y_1 = \mathbf{w}_1^T \mathbf{x} + b_1 \]

\[ y_2 = \mathbf{w}_2^T \mathbf{x} + b_2 \]

output:
\( i = \arg\max \{y_1, y_2\} \)
class \( i \)
Multi-class Option 1: Linear Regression/Perceptron: A Per-Class View (alternative)

\( y = \mathbf{w}^T \mathbf{x} + b \)

output:
if \( y > 0 \): class 1
else: class 2

\[ y_1 = [\mathbf{w}_1; \mathbf{w}_2]^T [\mathbf{x}; \mathbf{0}] + b_1 \]

\[ y_2 = [\mathbf{w}_1; \mathbf{w}_2]^T [\mathbf{0}; \mathbf{x}] + b_2 \]

\( i = \text{argmax} \{ y_1, y_2 \} \)

class \( i \)

Q: (For discussion) Why does this work?
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

(there can be others)

With C classes:

Train C different binary classifiers $\gamma_c(x)$

$\gamma_c(x)$ predicts 1 if $x$ is likely class $c$, 0 otherwise
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: **Build a one-vs-all (OvA) classifier**

Option 3: **Build an all-vs-all (AvA) classifier**

(there can be others)

With C classes:

Train C different binary classifiers

\[ \gamma_c(x) \]

\( \gamma_c(x) \) predicts 1 if \( x \) is likely class \( c \), 0 otherwise

To test/predict a new instance \( z \):

Get scores \( s^c = \gamma_c(z) \)

Output the max of these scores,

\( \hat{y} = \arg\max_c s^c \)
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: **Build an all-vs-all (AvA) classifier**

With C classes:

Train \( \binom{C}{2} \) different binary classifiers \( \gamma_{c_1,c_2}(x) \)

(there can be others)
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

(there can be others)

With C classes:

Train \( \binom{C}{2} \) different binary classifiers \( \gamma_{c_1, c_2}(x) \)

\( \gamma_{c_1, c_2}(x) \) predicts 1 if \( x \) is likely class \( c_1 \), 0 otherwise (likely class \( c_2 \))
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

(there can be others)

With C classes:

Train \( \binom{C}{2} \) different binary classifiers \( \gamma_{c_1,c_2}(x) \)

\( \gamma_{c_1,c_2}(x) \) predicts 1 if \( x \) is likely class \( c_1 \), 0 otherwise (likely class \( c_2 \))

To test/predict a new instance \( z \):

Get scores or predictions \( s^{c_1,c_2} = \gamma_{c_1,c_2}(z) \)
We’ve only developed binary classifiers so far...

**Option 1:** Develop a multi-class version

**Option 2:** Build a one-vs-all (OvA) classifier

**Option 3:** Build an all-vs-all (AvA) classifier

(there can be others)

With C classes:

Train \(_\binom{C}{2}\) different binary classifiers \(\gamma_{c_1,c_2}(x)\)

\(\gamma_{c_1,c_2}(x)\) predicts 1 if \(x\) is likely class \(c_1\), 0 otherwise (likely class \(c_2\))

To test/predict a new instance \(z\):

Get scores or predictions \(s^{c_1,c_2} = \gamma_{c_1,c_2}(z)\)

Multiple options for final prediction:

1. count # times a class \(c\) was predicted
2. margin-based approach
We’ve only developed binary classifiers so far…

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

(there can be others)

Q: (to discuss)

Why might you want to use option 1 or options OvA/AvA?

What are the benefits of OvA vs. AvA?
We’ve only developed binary classifiers so far...

Option 1: Develop a multi-class version

Option 2: Build a one-vs-all (OvA) classifier

Option 3: Build an all-vs-all (AvA) classifier

(there can be others)

Q: (to discuss)

Why might you want to use option 1 or options OvA/AvA?

What are the benefits of OvA vs. AvA?

What if you start with a balanced dataset, e.g., 100 instances per class?
Outline

Experimental Design: Rule 1

Multi-class vs. Multi-label classification

Evaluation

Regression Metrics
Classification Metrics
Regression Metrics

(Root) Mean Square Error

\[ RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \]
Regression Metrics

(Root) Mean Square Error \hspace{2cm} Mean Absolute Error

$$RMSE = \sqrt{\frac{1}{N} \sum_{i}^{N} (y_i - \hat{y}_i)^2}$$

$$MAE = \frac{1}{N} \sum_{i}^{N} |y_i - \hat{y}_i|$$
Regression Metrics

(Root) Mean Square Error

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i} (y_i - \hat{y}_i)^2}
\]

Mean Absolute Error

\[
MAE = \frac{1}{N} \sum_{i} |y_i - \hat{y}_i|
\]

Q: How can these reward/punish predictions differently?
Regression Metrics

(Root) Mean Square Error \hspace{1cm} Mean Absolute Error

\[ RMSE = \sqrt{\frac{1}{N} \sum_{i} (y_i - \hat{y}_i)^2} \]

\[ MAE = \frac{1}{N} \sum_{i} |y_i - \hat{y}_i| \]

Q: How can these reward/punish predictions differently?

A: RMSE punishes outlier predictions more harshly
Outline

Experimental Design: Rule 1

Multi-class vs. Multi-label classification

Evaluation

Regression Metrics

Classification Metrics
Training Loss vs. Evaluation Score

In training, compute loss to update parameters

Sometimes loss is a computational compromise
- surrogate loss

The loss you use might not be as informative as you’d like

  Binary classification: 90 of 100 training examples are +1, 10 of 100 are -1
Some Classification Metrics

Accuracy

Precision

Recall

AUC (Area Under Curve)

F1
## Classification Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th></th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selected/ Guessed</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Not selected/ not guessed</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Classification Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th>Selected/Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positive (TP)</td>
<td><img src="blue_circle.png" alt="Correct" /></td>
<td><img src="white_circle.png" alt="Guessed" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Not selected/not guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Classification Evaluation: 
the 2-by-2 contingency table

<table>
<thead>
<tr>
<th>Selected/Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positive</td>
<td>Correct (TP)</td>
<td>False Positive</td>
</tr>
<tr>
<td>Not selected/not guessed</td>
<td>Guessed</td>
<td>Correct (FP)</td>
</tr>
</tbody>
</table>

Classes/Choices
Classification Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th>Selected/ Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positive (TP)</td>
<td>Correct</td>
<td>Guessed</td>
<td></td>
</tr>
<tr>
<td>False Negative (FN)</td>
<td>Correct</td>
<td>Guessed</td>
<td></td>
</tr>
<tr>
<td>False Positive (FP)</td>
<td>Correct</td>
<td>Guessed</td>
<td></td>
</tr>
</tbody>
</table>
Classification Evaluation: the 2-by-2 contingency table

<table>
<thead>
<tr>
<th>Selected/Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positive</td>
<td>Correct</td>
<td>False Positive</td>
</tr>
<tr>
<td>(TP)</td>
<td>Guessed</td>
<td>(FP)</td>
</tr>
<tr>
<td>False Negative</td>
<td>Correct</td>
<td>True Negative</td>
</tr>
<tr>
<td>(FN)</td>
<td>Guessed</td>
<td>(TN)</td>
</tr>
</tbody>
</table>

Classes/Choices

- True Positive (TP)
- False Positive (FP)
- True Negative (TN)
- False Negative (FN)
Classification Evaluation: Accuracy, Precision, and Recall

**Accuracy**: % of items correct

\[
\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}
\]

<table>
<thead>
<tr>
<th>Selected/Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>True Positive</strong></td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td><strong>False Negative</strong></td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
<tr>
<td><strong>Not select/not guessed</strong></td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
## Classification Evaluation: Accuracy, Precision, and Recall

### Accuracy

**Accuracy**: % of items correct

\[
\frac{TP + TN}{TP + FP + FN + TN}
\]

### Precision

**Precision**: % of selected items that are correct

\[
\frac{TP}{TP + FP}
\]

### Table

<table>
<thead>
<tr>
<th>Selected/Guessed</th>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actually Correct</strong></td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td><strong>Actually Incorrect</strong></td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
**Classification Evaluation:**

**Accuracy:** % of items correct

\[
\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}
\]

**Precision:** % of selected items that are correct

\[
\text{Precision} = \frac{TP}{TP + FP}
\]

**Recall:** % of correct items that are selected

\[
\text{Recall} = \frac{TP}{TP + FN}
\]

<table>
<thead>
<tr>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selected/Guessed</strong></td>
<td></td>
</tr>
<tr>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td><strong>Not select/not guessed</strong></td>
<td></td>
</tr>
<tr>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
Classification Evaluation: Accuracy, Precision, and Recall

**Accuracy**: % of items correct

\[
\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}
\]

**Precision**: % of selected items that are correct

\[
\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}
\]

**Recall**: % of correct items that are selected

\[
\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}
\]

<table>
<thead>
<tr>
<th>Actually Correct</th>
<th>Actually Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selected/Guessed</strong></td>
<td><strong>Actualy Incorrect</strong></td>
</tr>
<tr>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td><strong>Not select/not guessed</strong></td>
<td>False Negative (FN)</td>
</tr>
<tr>
<td>True Negative (TN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
Precision and Recall Present a Tradeoff

Q: Where do you want your ideal model?
Precision and Recall Present a Tradeoff

Q: Where do you want your ideal model?

Q: You have a model that always identifies correct instances. Where on this graph is it?
Precision and Recall Present a Tradeoff

Q: You have a model that always identifies correct instances. Where on this graph is it?

Q: You have a model that only makes correct predictions. Where on this graph is it?

Q: Where do you want your ideal model?
Precision and Recall Present a Tradeoff

Q: Where do you want your ideal model?

Q: You have a model that always identifies correct instances. Where on this graph is it?

Q: You have a model that only make correct predictions. Where on this graph is it?
Precision and Recall Present a Tradeoff

Q: You have a model that always identifies correct instances. Where on this graph is it?

Q: You have a model that only makes correct predictions. Where on this graph is it?

Q: Where do you want your ideal model?

Idea: measure the tradeoff between precision and recall

Remember those hyperparameters: Each point is a differently trained/tuned model
Precision and Recall Present a Tradeoff

Q: Where do you want your ideal model?

Q: You have a model that always identifies correct instances. Where on this graph is it?

Q: You have a model that only make correct predictions. Where on this graph is it?

Idea: measure the tradeoff between precision and recall

Improve overall model: push the curve that way
Measure this Tradeoff:
Area Under the Curve (AUC)

AUC measures the area under this tradeoff curve

Min AUC: 0 😞
Max AUC: 1 😊
Measure this Tradeoff: Area Under the Curve (AUC)

AUC measures the area under this tradeoff curve

1. Computing the curve
   You need true labels & predicted labels with some score/confidence estimate
   
   Threshold the scores and for each threshold compute precision and recall

Min AUC: 0 😞
Max AUC: 1 😊
Measure this Tradeoff: Area Under the Curve (AUC)

AUC measures the area under this tradeoff curve

1. **Computing the curve**
   - You need true labels & predicted labels with some score/confidence estimate
   - Threshold the scores and for each threshold compute precision and recall

2. **Finding the area**
   - How to implement: trapezoidal rule (& others)

**In practice**: external library like the sklearn.metrics module

- Min AUC: 0 😞
- Max AUC: 1 😊
Measure A Slightly Different Tradeoff: ROC-AUC

AUC measures the area under this tradeoff curve

1. Computing the curve
   You need true labels & predicted labels with some score/confidence estimate
   Threshold the scores and for each threshold compute metrics

2. Finding the area
   How to implement: trapezoidal rule (& others)

In practice: external library like the sklearn.metrics module

Main variant: ROC-AUC
Same idea as before but with some flipped metrics

Min ROC-AUC: 0.5 😞
Max ROC-AUC: 1 😊
A combined measure: $F$

Weighted (harmonic) average of Precision & Recall

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}}$$
A combined measure: F

Weighted (harmonic) average of Precision & Recall

\[ F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R} \]

algebra (not important)
A combined measure: $F$

Weighted (harmonic) average of Precision & Recall

$$F = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R}$$

Balanced F1 measure: $\beta=1$

$$F_1 = \frac{2 \times P \times R}{P + R}$$
P/R/F in a Multi-class Setting: Micro- vs. Macro-Averaging

If we have more than one class, how do we combine multiple performance measures into one quantity?

**Macroaveraging**: Compute performance for each class, then average.

**Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.
P/R/F in a Multi-class Setting: Micro- vs. Macro-Averaging

**Macroaveraging**: Compute performance for each class, then average.

\[
\text{macroprecision} = \sum_c \frac{TP_c}{TP_c + FP_c} = \sum_c \text{precision}_c
\]

**Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.

\[
\text{microprecision} = \frac{\sum_c TP_c}{\sum_c TP_c + \sum_c FP_c}
\]
P/R/F in a Multi-class Setting: Micro- vs. Macro-Averaging

**Macroaveraging**: Compute performance for each class, then average.

\[
\text{macroprecision} = \sum_c \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c} = \sum_c \text{precision}_c
\]

**Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.

\[
\text{microprecision} = \frac{\sum_c \text{TP}_c}{\sum_c \text{TP}_c + \sum_c \text{FP}_c}
\]

when to prefer the macroaverage?

when to prefer the microaverage?
**Micro- vs. Macro-Averaging: Example**

<table>
<thead>
<tr>
<th>Class 1</th>
<th></th>
<th>Class 2</th>
<th></th>
<th>Micro Ave. Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Truth : yes</td>
<td>Truth : no</td>
<td></td>
<td>Truth : yes</td>
</tr>
<tr>
<td>Classifier: yes</td>
<td>10</td>
<td>10</td>
<td>Classifier: yes</td>
<td>90</td>
</tr>
<tr>
<td>Classifier: no</td>
<td>10</td>
<td>970</td>
<td>Classifier: no</td>
<td>10</td>
</tr>
</tbody>
</table>

Macroaveraged precision: \((0.5 + 0.9)/2 = 0.7\)

Microaveraged precision: \(100/120 = .83\)

Microaveraged score is dominated by score on frequent classes
Some Classification Metrics

Accuracy

Precision

Recall

AUC (Area Under Curve)

F1
Outline

Experimental Design: Rule 1

Multi-class vs. Multi-label classification

Evaluation

Regression Metrics

Classification Metrics