Lecture 20: How to Build a Classifier (LR) from Dataset, From Logistic Regression to Conditional Random Fields and Graphical Models in General, VC Dimensions, Neural Networks

Instructor: Prof. Ganesh Ramakrishnan

Illustrating Logistic Regression on Travel Mode Data

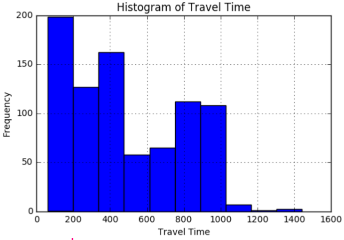
- Number of observations: 840 Observations On 4 Modes for 210 Individuals.
- Number of variables: 8
- Variable name definitions::
  1. individual = 1 to 210
  2. mode = 1 - air, 2 - train, 3 - bus, 4 - car
  3. choice = 0 - no, 1 - yes
  4. ttme = terminal waiting time for plane, train and bus (minutes); 0 for car.
  5. invc = in vehicle cost for all stages (dollars).
  6. invt = travel time (in-vehicle time) for all stages (minutes).
  7. gc = generalized cost measure:invc+(invt*value of travel time savings) (dollars).
  8. hinc = household income ($1000s).
  9. psize = traveling group size in mode chosen (number).

```
from sklearn.linear_model import LogisticRegression
from sklearn.cross_validation import train_test_split
from sklearn import metrics
from sklearn.cross_validation import cross_val_score
# load dataset
#http://statsmodels.sourceforge.net/0.6.0/datasets/generated/modechoice.html
dta = sm.datasets.modechoice.load_pandas().data
# show plots in the notebook
%matplotlib inline
# histogram of education
#print(sm.datasets.modechoice.load_pandas())
#dta.educ.hist()
dta.groupby('mode').mean()
```

	individual	choice	ttme	invc	invt	gc	hinc	psize
mode								
1.0	105.5	0.276190	61.009524	85.252381	133.709524	102.647619	34.547619	1.742857
2.0	105.5	0.300000	35.690476	51.338095	608.285714	130.200000	34.547619	1.742857
3.0	105.5	0.142857	41.657143	33.457143	629.461905	115.257143	34.547619	1.742857
4.0	105.5	0.280952	0.000000	20.995238	573.204762	95.414286	34.547619	1.742857

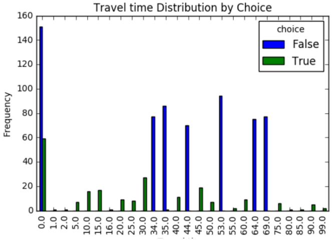
In [34]:
```python
dta.invt.hist()
plt.title('Histogram of Travel Time')
plt.xlabel('Travel Time')
plt.ylabel('Frequency')
```

Out[34]: <matplotlib.text.Text at 0x1b974b6c2b0>



In [43]:
```python
# barplot of travel time grouped by Choice
pd.crosstab(dta.ttme, dta.choice.astype(bool)).plot(kind='bar')
plt.title('Travel time Distribution by Choice')
plt.xlabel('Travel time')
plt.ylabel('Frequency')
```

Out[43]: <matplotlib.text.Text at 0x1b9785421d0>



upyter Classification 1 Last Checkpoint: 6 hours ago (autosaved)

e Edit View Insert Cell Kernel Widgets Help

Code CellToolbar

In [63]:
```
# create dataframes with an intercept column and dummy variables for
# occupation and occupation_husb
y, X = dmatrices('choice ~ ttme + invc + invt + gc + psize',
                  dta, return_type="dataframe")
print(X.columns)
# flatten y into a 1-D array
y = np.ravel(y)
```

```
Index(['Intercept', 'ttme', 'invc', 'invt', 'gc', 'psize'], dtype='object')
```

In [64]:
```
# instantiate a logistic regression model, and fit with X and y
model = LogisticRegression()
model = model.fit(X, y)

# check the accuracy on the training set
model.score(X, y)
```

Out[64]:
```
0.80119047619047623
```

In [65]:
```
# what percentage got their choice?
y.mean()
```

Out[65]:
```
0.25
```

In [66]:
```
# evaluate the model by splitting into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
model2 = LogisticRegression()
model2.fit(X_train, y_train)
```

Out[66]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

Let us understand the results from our simulation on the test data:

```
In [70]: print(metrics.confusion_matrix(y_test, predicted))
         print(metrics.classification_report(y_test, predicted))
```

```
[[193   7]
 [ 39  13]]
```

→ Confusion matrix

	precision	recall	f1-score	support
0.0	0.83	0.96	0.89	200
1.0	0.65	0.25	0.36	52
avg / total	0.79	0.82	0.78	252

$$F_1[0] = \frac{2\,Pr[0]\,Re[0]}{Pr[0] + Re[0]}$$

	Predicted #0's	#1's
Actual #0's	193	7
Actual #1's	39	13

True negatives (TNs)

False positives (FPs)

(FN) False Negatives

True positives (TPs)

$$Pr[0] = \frac{TN}{TN+FN}$$

$$Re[0] = \frac{TN}{TN+FP}$$

h/w! $Pr[1]$ & $Re[1]$?

Aside: Computing class label in Logistic Regression:

$> 0.5 \Rightarrow$ class 1

$< 0.5 \Rightarrow$ class 0

Most common (not best)

$$\text{Accuracy} = \frac{\text{Diagonal sum}}{\text{Sum of all elements}}$$

$$= \frac{TP + TN}{TP + TN + FP + FN}$$

F_1 gives a lot of imp to $\min(Pr, Re) \leq GM \leq AM$

Accuracy gives more imp to $\max(Pr, Re)$

Logistic Regression Generalized to CRF

1. Multi-class LR: $c \in [1 \dots K]$ has weight vector $[w_{c,1} \dots w_{c,p}]$

$$\Pr(y = c \mid x) = \frac{e^{-w_c^T \phi(\mathbf{x})}}{\sum_{k=1}^{K} e^{-w_k^T \phi(\mathbf{x})}} = \frac{e^{-\tilde{w}^T \tilde{\phi}(x,c)}}{\sum_{k=1}^{K} e^{-\tilde{w}^T \tilde{\phi}(x,k)}}$$

WHY?

DESIRABLE

c_1 c_2 c_3 c_4 c_5 . . . . .

I do not mind the extra class tomorrow.

verb

$\tilde{\phi} \equiv$ words

$\underbrace{\text{not} \qquad \text{mind}}$

$c_3 = AD$ $\quad c_4 = \text{verb}$

$c'_6 = PPN$ $\quad c'_7 = \text{noun}$

$\underbrace{\text{my} \qquad \text{mind}}$

$\tilde{\phi}$ labels

$(PPN, Noun) = 1$

Correlations between outputs

An extra class will blow my mind off!

noun

c'_1 c'_2 c'_3 c'_4 c'_5 c'_6 c'_7

[1] http://www.tzi.de/~edelkamp/lectures/ml/scripts/loglinearcrfs.pdf

Logistic Regression Generalized to CRF

1. Multi-class LR: $c \in [1 \ldots K]$ has weight vector $[w_{c,1} \ldots w_{c,p}]$

How!

$$\Pr(y = c \mid x) = \frac{e^{-w_c^T \phi(\mathbf{x})}}{\sum_{k=1}^{K} e^{-w_k^T \phi(\mathbf{x})}} = \frac{e^{-\tilde{w}^T \phi(\mathbf{x}, y=c)}}{Z(\mathbf{x}, \tilde{w})}$$

where $\tilde{w} = [\overbrace{w_{1,1} \ldots w_{1,p}}^{w_1}, \ldots \overbrace{w_{c,1} \ldots w_{c,p}}^{w_c}, \ldots \overbrace{w_{K,1} \ldots w_{K,p}}^{w_K}]$ and $\phi(\mathbf{x}, y) = [\underbrace{\delta(y,1)\phi(\mathbf{x})}_{\phi(x) \text{ if } y=1}, \ldots, \underbrace{\delta(y,c)\phi(\mathbf{x})}_{\phi(x) \text{ if } y=c \text{ \& others 0's}} \ldots \delta(y,K)\phi(\mathbf{x})]$ and $\delta(a,b) = 1$ if $a = b$ and $= 0$ otherwise

2. Extended to non-iid inference in Conditional Random Fields[1] with $\mathbf{x} = [\mathbf{x}^{(1)} \ldots \mathbf{x}^{(n)}]$ and $\mathbf{y} = [y^{(1)} \ldots y^{(n)}]$:

I do mind tomorrow

PN VB

$$\Pr(Y = [y^{(1)} \ldots y^{(n)}]) = \frac{e^{-w^T \phi(x^1 \ldots x^n, y \ldots y^n)}}{Z(x^1 \ldots x^n, w)}$$

Generalize this expression

[1] http://www.tzi.de/~edelkamp/lectures/ml/scripts/loglinearcrfs.pdf

Logistic Regression Generalized to CRF

1. Multi-class LR: $c \in [1 \ldots K]$ has weight vector $[w_{c,1} \ldots w_{c,p}]$

$$\Pr(y = c \mid x) = \frac{e^{-w_c^T \phi(\mathbf{x})}}{\sum_{k=1}^{K} e^{-w_k^T \phi(\mathbf{x})}} = \frac{e^{-\tilde{w}^T \phi(\mathbf{x}, y=c)}}{Z(\mathbf{x}, \tilde{w})}$$

   where $\tilde{w} = [w_{1,1} \ldots w_{1,p}, \ldots w_{c,1} \ldots w_{c,p}, \ldots w_{K,1} \ldots w_{K,p}]$ and $\phi(\mathbf{x}, y) = [\delta(y,1)\phi(\mathbf{x}), \ldots, \delta(y,c)\phi(\mathbf{x}) \ldots \delta(y,K)\phi(\mathbf{x})]$ and $\delta(a,b) = 1$ if $a = b$ and $= 0$ otherwise

2. Extended to non-iid inference in Conditional Random Fields[1] with $\mathbf{x} = [\mathbf{x}^{(1)} \ldots \mathbf{x}^{(n)}]$ and $\mathbf{y} = [y^{(1)} \ldots y^{(n)}]$:

$$\Pr(\mathbf{y} \mid \mathbf{x}) = \frac{e^{-\mathbf{w}^T \phi(\mathbf{x}, \mathbf{y})}}{Z(\mathbf{x}, \mathbf{w})}$$

Q: Can we avoid enumeration of K^n possible $[y^1 \ldots y^n]$ since each $y^i \in \{c_1 \ldots c_K\}$

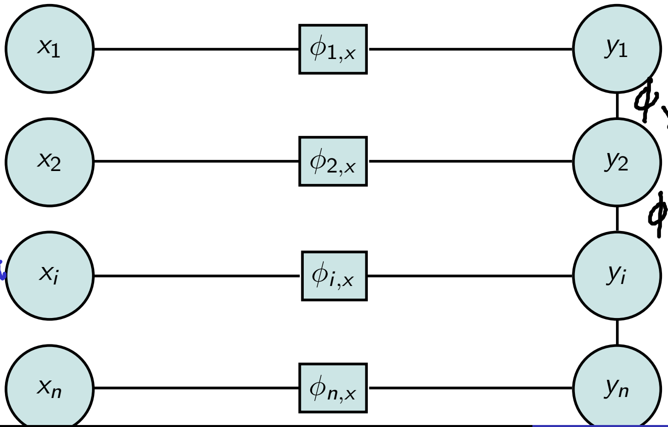
[1] http://www.tzi.de/~edelkamp/lectures/ml/scripts/loglinearcrfs.pdf

Conditional Random Fields (Linear)[a]

[a] CRF is an instance of Graphical Models (detailed notes at https://www.cse.iitb.ac.in/~cs725/notes/classNotes/misc/CaseStudyWithProbabilisticModels.pdf

} Extra reading

inputs | $x-$potentials | classes & $y-$potentials $\phi_{i,y}$



I

do

not

mind

$\phi_{y_1 y_2}(\cdot)$

$\phi_{y_2 y_3}(\cdot)$

$\{C_1 .. C_k\}$

$\{C_1 .. C_k\}$

$\{C_1 .. C_k\}$

$e^{-w^T[\phi_{xy}(x^i,y^i)\,\phi_{yy}(y^i,y^{i-1})\,\phi_{yy}(y^i,y^{i+1})]}$

Pairwise interactions make Polytime decision making possible

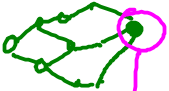
Non-linear perceptron?

- Kernelized perceptron: $f(x) = sign\left(\sum_i \alpha_i y_i K(x, x_i) + b\right)$
    - INITIALIZE: α=zeroes()
    - REPEAT: for $< x_i, y_i >$
        - If $sign\left(\sum_j \alpha_j y_j K(x_j, x_i) + b\right) \neq y_i$
        - then, $\alpha_i = \alpha_i + 1$
        - endif

Coming up with a suitable kernel can be challenging!

0/1 ← → sigmoidal

- Neural Networks: Cascade of layers of perceptrons giving you non-linearity
    - $sign\left((w^*)^T \phi(x)\right)$ replaced by $g\left((w^*)^T \phi(x)\right)$ where $g(s)$ is a
        1. step function: $g(s) = 1$ if $s \in [\theta, \infty)$ and $g(s) = 0$ otherwise OR
        2. sigmoid function: $g(s) = \frac{1}{1+e^{-s}}$ with possible thresholding using some θ (such as $\frac{1}{2}$).
        3. Rectified Linear Unit (ReLU): $g(s) = max(0, s)$: A most popular activation function
        4. Softplus: $g(s) = \ln\left(1 + e^s\right)$



Options 2, 3 and 4 have the thresholding step deferred. Threshold changes as bias is changed.

Thresholding postponed to final layer/node.

Measure for Linear non-separability?

- **Aspect 1: Number of functions that can be represented** Recall from Tutorial 6, Problem 1: Given n boolean variables how many of 2^{2^n} boolean functions can be represented by a perceptron? Ans: For 2 it is 14, for 3 it is 104, for 4 it is 1882

S_n = Count of #fns that can seperate n pts

Non sep measure = $1 - \frac{S_n}{F_n}$

F_n = total # of fns = 2^{2^n}

But how abt non-boolean case?

Measure for Linear non-separability?

- **Aspect 1: Number of functions that can be represented** Recall from Tutorial 6, Problem 1: Given n boolean variables how many of 2^{2^n} boolean functions can be represented by a perceptron? Ans: For 2 it is 14, for 3 it is 104, for 4 it is 1882
- **Aspect 2: Cardinality of largest set of points that can be shattered** VC (VapnikChervonenkis) dimension $\Rightarrow$ A measure of the richness of a space of functions that can be learned by a statistical classification algorithm.
  - A classification function $f(\mathbf{w})$ is said to shatter a set of data points $(x_1, x_2, \ldots, x_n)$ if, for all assignments of labels to those points, there exists a $\mathbf{w}$ such that $f(\mathbf{w})$ makes no errors when evaluating that set of data points.
  - Cardinality of the largest set of points that $f(\mathbf{w})$ can **shatter** is its VC-dimension.
  - Eg: For f as a threshold interval, VC = 1

Perceptron with bias only

$\theta \equiv w$

For $n=1$, if $y_1 = +ve$ choose $\theta = x_1 - 1/2$
if $y_1 = -ve$ choose $\theta = x_1 + 1/2$
$\exists\ \theta \equiv w$ that correctly classifies

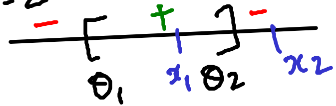
$x \geq \theta \Rightarrow y = 1$
$x < \theta \Rightarrow y = 0$

For $n=2$,
$x_1 = 1 \quad y_1 = 1$
$x_2 = 2 \quad y_2 = -1$
No θ exists!
Desired: $y_2 \geq y_1$

Measure for Linear non-separability?

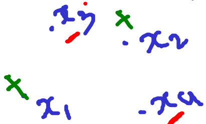
- **Aspect 1: Number of functions that can be represented** Recall from Tutorial 6, Problem 1: Given n boolean variables how many of 2^{2^n} boolean functions can be represented by a perceptron? Ans: For 2 it is 14, for 3 it is 104, for 4 it is 1882
- **Aspect 2: Cardinality of largest set of points that can be shattered** VC (VapnikChervonenkis) dimension $\Rightarrow$ A measure of the richness of a space of functions that can be learned by a statistical classification algorithm.
  - A classification function $f(\mathbf{w})$ is said to shatter a set of data points $(x_1, x_2, \ldots, x_n)$ if, for all assignments of labels to those points, there exists a $\mathbf{w}$ such that $f(\mathbf{w})$ makes no errors when evaluating that set of data points.
  - Cardinality of the largest set of points that $f(\mathbf{w})$ can **shatter** is its VC-dimension.
  - Eg: For f as a threshold interval, VC dimension $= 1$
  - Eg: For f as an interval classifier, $VC = 2$

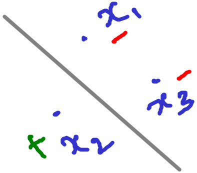
$n = 2$, $x_1 = 1$ $y_1 = 1$
$x_2 = 2$ $y_2 = -1$



Measure for Linear non-separability?

- **Aspect 1: Number of functions that can be represented** Recall from Tutorial 6, Problem 1: Given n boolean variables how many of 2^{2^n} boolean functions can be represented by a perceptron? Ans: For 2 it is 14, for 3 it is 104, for 4 it is 1882
- **Aspect 2: Cardinality of largest set of points that can be shattered** VC (VapnikChervonenkis) dimension $\Rightarrow$ A measure of the richness of a space of functions that can be learned by a statistical classification algorithm.
  - A classification function $f(\mathbf{w})$ is said to shatter a set of data points $(x_1, x_2, \ldots, x_n)$ if, for all assignments of labels to those points, there exists a $\mathbf{w}$ such that $f(\mathbf{w})$ makes no errors when evaluating that set of data points.
  - Cardinality of the largest set of points that $f(\mathbf{w})$ can **shatter** is its VC-dimension.
  - Eg: For f as a threshold interval, VC dimension $= 1$
  - Eg: For f as an interval classifier, VC dimension $= 2$
  - Eg: For f as linear classifier (in 2 dimensions), VC=3





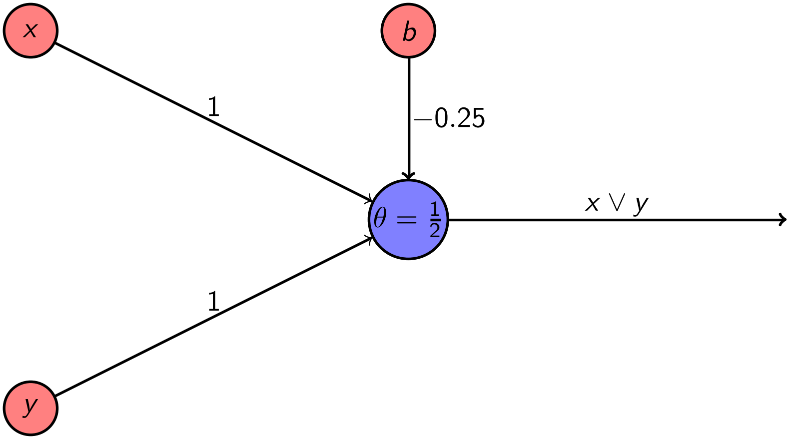
Measure for Linear non-separability?

- **Aspect 1: Number of functions that can be represented** Recall from Tutorial 6, Problem 1: Given n boolean variables how many of 2^{2^n} boolean functions can be represented by a perceptron? Ans: For 2 it is 14, for 3 it is 104, for 4 it is 1882
- **Aspect 2: Cardinality of largest set of points that can be shattered** VC (VapnikChervonenkis) dimension $\Rightarrow$ A measure of the richness of a space of functions that can be learned by a statistical classification algorithm.
  - A classification function $f(\mathbf{w})$ is said to shatter a set of data points $(x_1, x_2, \ldots, x_n)$ if, for all assignments of labels to those points, there exists a $\mathbf{w}$ such that $f(\mathbf{w})$ makes no errors when evaluating that set of data points.
  - Cardinality of the largest set of points that $f(\mathbf{w})$ can **shatter** is its VC-dimension.
  - Eg: For f as a threshold interval, VC dimension $= 1$
  - Eg: For f as an interval classifier, VC dimension $= 2$
  - Eg: For f as linear classifier (in 2 dimensions), VC dimension $= 3$
  - Eg: For f as linear classifier (in $\Re^n$), VC= n+1

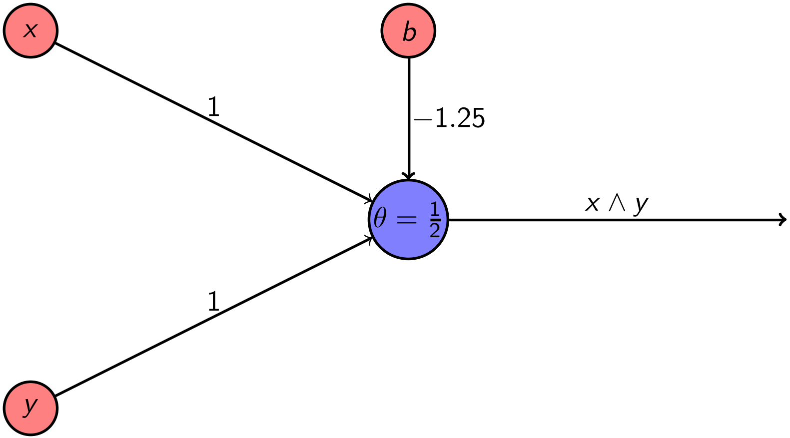
Measure for Linear non-separability?

- **Aspect 1: Number of functions that can be represented** Recall from Tutorial 6, Problem 1: Given n boolean variables how many of 2^{2^n} boolean functions can be represented by a perceptron? Ans: For 2 it is 14, for 3 it is 104, for 4 it is 1882
- **Aspect 2: Cardinality of largest set of points that can be shattered** VC (VapnikChervonenkis) dimension $\Rightarrow$ A measure of the richness of a space of functions that can be learned by a statistical classification algorithm.
    - A classification function $f(\mathbf{w})$ is said to shatter a set of data points $(x_1, x_2, \ldots, x_n)$ if, for all assignments of labels to those points, there exists a $\mathbf{w}$ such that $f(\mathbf{w})$ makes no errors when evaluating that set of data points.
    - Cardinality of the largest set of points that $f(\mathbf{w})$ can **shatter** is its VC-dimension.
    - Eg: For f as a threshold interval, VC dimension $= 1$
    - Eg: For f as an interval classifier, VC dimension $= 2$
    - Eg: For f as linear classifier (in 2 dimensions), VC dimension $= 3$
    - Eg: For f as linear classifier (in $\Re^n$), VC dimension $=$ n$+$ 1
    - Eg: For f as a neural network with sigmoid function, V nodes and E edges, then the VC dimension is at least $\Omega(|E|^2)$ and at most $O(|E|^2 \cdot |V|^2)$

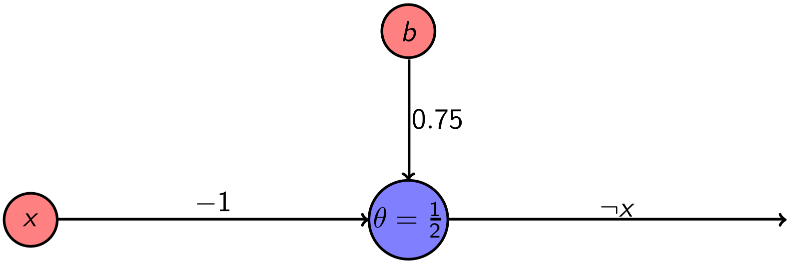
OR using (step) perceptron



AND using (step) perceptron



NOT using perceptron



Feed-forward Neural Nets