Exam 1 preview

Glenn Bruns
CSUMB

Coverage of exam

All material including ensemble methods.

- linear algebra
- training models
- support vector machines
- ensemble learning

Structure of exam

1. Concepts and theory (20 mins)
   - conceptual questions
   - questions about the math
   - paper and pencil – no notes or other resources
2. Practical (40 mins)
   - add your code to starter iPython notebook
   - use any resources you like

How to prepare

- Note learning outcomes at the front of each slide deck
  - ask yourself if you can do these things
- Practice on lab and homework problems
- Don’t passively review lecture slides
  - actively review by writing test questions
  - make flash cards for yourself, and use them

Practical part of exam

Using Numpy, Scikit-Learn, Pandas, Seaborn:

- create exploratory plots
- preprocess (impute, categorical → numeric, scale)
  - pipeline not needed
- apply machine learning algorithms
  - basics, SVM, ensemble methods
  - be able to tune hyperperameters
  - be able to diagnose results

Linear algebra: vectors

- Vector operations
  - addition, multiplication by scalar
  - norm, dot product
  - properties of these operations
- Special vectors: zero, unit, normalized

You should be able to perform the operations by hand

Question

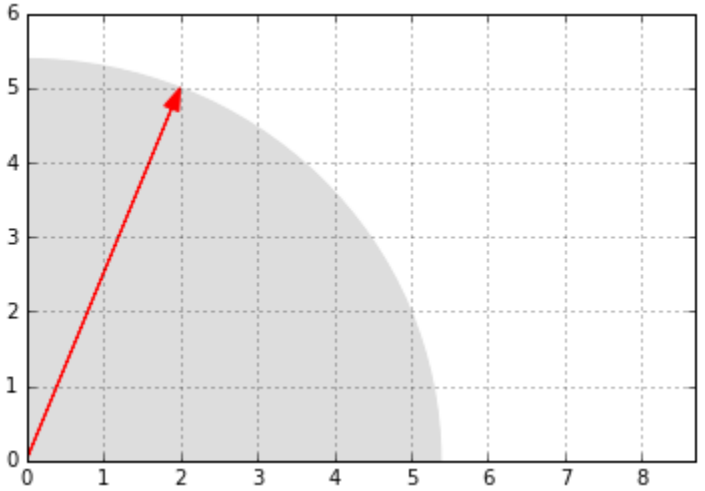
Add these vectors:

$$\begin{pmatrix} 1 \\ 2 \\ 6 \end{pmatrix} \begin{pmatrix} -2 \\ 0 \\ 5 \end{pmatrix}$$

Question

(T/F) The norm $\|u\|$ of a vector u is a vector.

False. The norm of a vector is a scalar that gives the vector's length.



The norm of $\binom{2}{5}$ is about 5.4

$$\sqrt{2^2 + 5^2} \cong 5.38$$

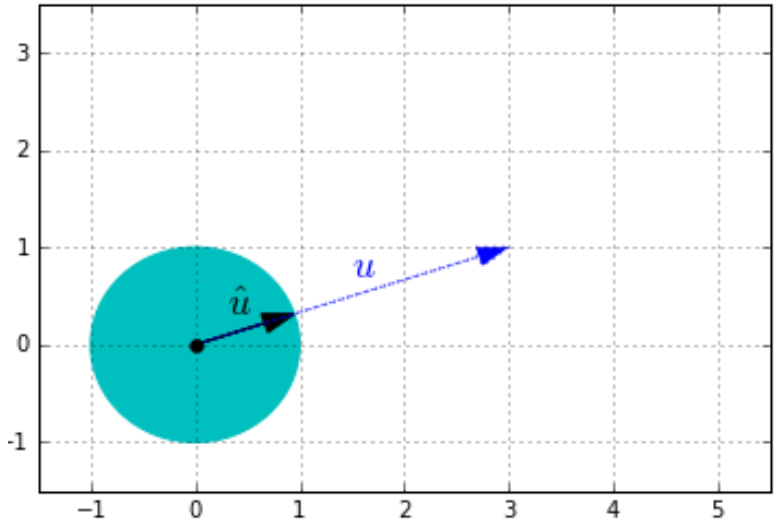
Question

(true/false)

Vectors u and $\hat{u}$ have the same direction.

true

$\hat{u}$ is the normalized version of u – a unit vector in the same direction as u



normalized vector

Question

(yes/no)

Is (vector) dot product commutative?

yes

$$\begin{pmatrix} 3 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 5 \end{pmatrix} = 3(2) + 1(5)$$

Question

Without visualizing or drawing them, check you check whether $\binom{3}{1}$ and $\binom{-1}{3}$ are perpendicular?

We can test using the dot product

$$\binom{3}{1} \cdot \binom{-1}{3} = 0$$

So the vectors are perpendicular.

Question

Without visualizing or drawing them, check you check whether $\binom{3}{1}$ and $\binom{-1}{3}$ are perpendicular?

We can test using the dot product

$$\binom{3}{1} \cdot \binom{-1}{3} = 0$$

So the vectors are perpendicular.

Dot product of two vectors

Definition: (dot product also known as "inner product")

$$u \cdot v = \sum_i u_i \times v_i$$

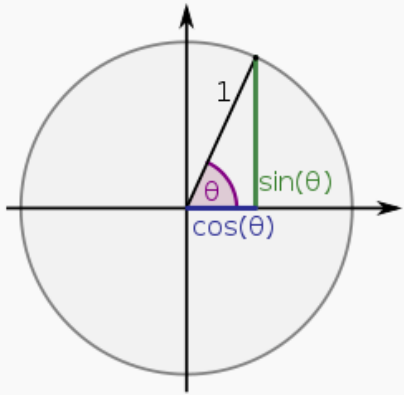
Example:

$$\binom{3}{1} \cdot \binom{2}{5} = 3(2) + 1(5) = 11$$

An alternative definition is used in physics:

$$u \cdot v = \|u\| \times \|v\| \times \cos(\theta)$$

where θ is the angle between u and v




```
# dot product with NumPy
np.dot(u,v)
# alternative
u.dot(v)
```

Commutative: $u \cdot v = v \cdot u$

Associative: ?

Associates with scalar multiplication:

$$k(u \cdot v) = ku \cdot kv$$

Distributes over vector addition:

$$u \cdot (v + w) = (u \cdot v) + u \cdot w)$$

Linear algebra: matrices

- Matrix operations
  - addition, multiplication by scalar
  - matrix multiplication (and when its possible)
  - properties of these operations
- Special matrices: square, diagonal, identity

Note that the operations of addition and multiplication by scalar are very similar for vectors and matrices

The have the same properties (because they're both examples of vector spaces).

Question

True/False

(b + c)A = bA + cA

True. This should be intuitively obvious.

$$(b+c)\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix} = \begin{pmatrix}(b+c)1 & (b+c)2\\ (b+c)3 & (b+c)4\end{pmatrix}$$

Question

True/False

An identity matrix is an upper triangular matrix.

True.

$$\begin{pmatrix} 6 & 4 & 2 \\ 0 & 8 & 7 \\ 0 & 0 & 3 \end{pmatrix}$$

upper triangular

(square, and values below main diagonal all 0)

Question

True/False

Only square matrices can be multiplied.

False.

# of cols in first matrix must equal # of rows in second matrix

m x **n** **n** x p → m x p

Question

True/False

$$A(B + C) = AB + AC$$

True.

Question

True/False

From every pair u, v of distinct, non-0 2D vectors, it is possible to get any other 2D vector through linear combination

False.

$$b\begin{pmatrix}1\\2\end{pmatrix} + c\begin{pmatrix}2\\4\end{pmatrix} = \begin{pmatrix}b+2c\\2b+4c\end{pmatrix} = \begin{pmatrix}x\\2x\end{pmatrix}$$

Properties of addition and scal. mult.

vectors

$$u + w = w + u$$

$$u + (w + v) = (u + w) + v$$

$$(bc)u = b(cu)$$

$$c(u + w) = cu + cw$$

$$(b + c)u = bu + cu$$

matrices

$$A + B = B + A$$

$$A + (B + C) = (A + B) + C$$

$$(bc)A = b(cA)$$

$$c(A + B) = cA + cB$$

$$(b + c)A = bA + cA$$

b, c – scalars
u, v, w – vectors
A, B, C - matrices

Question

True or False

A matrix can have two distinct inverses.

False.

Suppose B,C are distinct inverses of A.

$B(AC) = (BA)C$

$BI = IC$

$B = C$

Question

True or False

Every matrix can be transposed.

True.

LA: matrix inversion and transpose

- Matrix multiplication properties
- Matrix transpose and its properties
- Matrix inversion, invertible matrices
- Singular matrices
- Matrices as functions/transformers

You should be able to determine if a 2 x 2 matrix is invertible

Is this matrix invertible?

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

Is there a square matrix B such that

$$AB = I \text{ ?}$$

A square matrix that is not invertible is called singular (or degenerate).

Notes:

- non-square matrices don't have inverses
- $(A^{-1})^{-1} = A$ (when A is invertible)

```
A = np.array([
        [1, 0],
        [0, 0]
    ])
LA.inv(A)
```

```
In [5]: LA.inv(A)
Out[103]: ...
LinAlgError: Singular matrix
```

LA: singular value decomposition

- determinants and relation to singularity
- composing transformations
- singular value decomposition (SVD)
- eigenvalues and eigenvectors

I don’t expect you to remember how to compute determinants

For this material just focus on main points

Question

Yes/No

Is $\begin{pmatrix}1\\2\end{pmatrix}$ an eigenvector of $\begin{pmatrix}1 & 0\\2 & 1\end{pmatrix}$?

No.

$$\begin{pmatrix}1 & 0\\2 & 1\end{pmatrix}\begin{pmatrix}1\\2\end{pmatrix} = \begin{pmatrix}1\\3\end{pmatrix}$$

and $\begin{pmatrix}1\\2\end{pmatrix}$ and $\begin{pmatrix}1\\3\end{pmatrix}$ are not related by a scalar.

Singular value decomposition (SVD)

Integers can be "decomposed" into a product of primes.

Matrices can be "decomposed" into the product of three simple matrices.

Any $m \times n$ matrix A can be decomposed like this:

$$A = U\Sigma V^T$$

where

- U is a rotation matrix (an $m \times m$ orthogonal matrix)
- Σ is a scaling & projecting matrix (an $m \times n$ diagonal matrix)
- V^T is a rotation matrix (an $n \times n$ orthogonal matrix)

A square matrix H is orthogonal if its inverse is the same as its transpose:

$$H^{-1} = H^T$$

as a result:

$$HH^T = H^T H = 1$$

Training models: gradient descent

- ways to find max or min of a function
  - grid search, derivatives, gradient descent
- gradient descent
- gradient descent for multi-variable functions
- linear regression as an optimization problem
- pros/cons of closed form solution for linear regression

Question

What is the approximate value of $f'(2)$?

x	f(x)
1.9	9.31
2.1	10.71

$$\frac{10.71 - 9.31}{0.2}$$

Batch gradient descent

```
eta = 0.1  # learning rate
n_iterations = 1000
m = 100

theta = np.random.randn(2,1)  # random initialization

for iteration in range(n_iterations):
    gradients = 2/m * X_b.T.dot(X_b.dot(theta) - y)
    theta = theta - eta * gradients
```

theta that maximizes negative cost =

theta that minimizes cost

source: Géron, Hands-On Machine Learning text

TM: stochastic gradient descent

- batch gradient descent
- stochastic gradient descent
- mini-batch gradient descent
- pros/cons of gradient descent for linear regression

We started thinking about cost functions in machine learning, and their relationship to gradient descent.

Stochastic gradient descent

```
n_epochs = 50
t0, t1 = 5, 50  # learning schedule hyperparameters

def learning_schedule(t):
    return t0 / (t + t1)

theta = np.random.randn(2,1)  # random initialization

for epoch in range(n_epochs):
    for i in range(m):
        random_index = np.random.randint(m)
        xi = X_b[random_index:random_index+1]
        yi = y[random_index:random_index+1]
        gradients = 2 * xi.T.dot(xi.dot(theta) - yi)
        eta = learning_schedule(epoch * m + i)
        theta = theta - eta * gradients
```

source: Géron, Hands-On Machine Learning text

Recall: ML as optimization

steps	example: linear regression
define a **model**, with parameters, that will be used to make predictions	$\hat{y} = \theta^T \cdot \mathbf{x}$
define a **cost function** to explain what it means for the model to fit the data well	$MSE(\mathbf{X}, \theta) = \frac{1}{m} \sum_{i=1}^{m} \left(\theta^T \cdot \mathbf{x}^{(i)} - y^{(i)}\right)^2$
if possible, find the **partial derivatives of the cost function**	$\frac{\partial}{\partial \theta_j} MSE(\mathbf{X}, \theta) = \frac{2}{m} \sum_{i=1}^{m} \left(\theta^T \cdot x^{(i)} - y^{(i)}\right) x_j^{(i)}$
fit the model to your training data by finding the parameters that **minimize the cost function, using gradient descent**	

TM: regularization

- regularization concept
- regularization and bias/variance
- regularization in optimization
- regularization methods in linear regression:
  - ridge regression, the lasso, early stopping
- regularization and feature selection

Correction:
- 'elastic net' is a combination of ridge regression and the lasso
- a summary side incorrectly said 'early stopping' instead of 'elastic net'

Lasso regression

Very similar to ridge regression. This is the ridge regression cost function:

$$MSE(\theta) + \frac{\alpha}{2} \sum_{i=1}^{n} \theta_i^2$$

This is the lasso regression cost function:

$$MSE(\theta) + \frac{\alpha}{2} \sum_{i=1}^{n} |\theta|$$

The change is small. Surprisingly, lasso tends to zero the weights of less important features.

TM: logistic regression

- review of logistic regression
- logistic regression as an optimization function
  - focus on cost function for linear regression
- how to build multi-class classifiers from binary classifiers: one-vs-one and one-vs-all
- “native” multi-class logistic regression (aka “softmax regression”)

Cost function for entire training set

Cost for a single training example:

$$c(\theta) = \begin{cases} -\log(\hat{p}) & \text{if } y = 1 \\ -\log(1 - \hat{p}) & \text{if } y = 0 \end{cases}$$

Cost for training set is the average cost per training example:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(\hat{p}^{(i)}\right) + \left(1 - y^{(i)}\right)\log(1 - \hat{p}^{(i)})\right]$$

"log loss"

m : number of training examples
$\hat{p}^{(i)}$: predicted probability of ith training example
$y^{(i)}$: label (0 or 1) or ith training example

Support vector machines: concept

- street, margin, support vector, linearly separable
- hard-margin classifier vs. soft-margin classifier
- tuning parameter C
- hard and soft-margin classification as an optimization problem

Soft Margin Classifier: general form

This optimization problem balances the size of the margin and sum of the margin violations.

$$\underset{\mathbf{w},b,\zeta}{\text{minimize}} \quad \frac{1}{2}\mathbf{w}^T \cdot \mathbf{w} + C\sum_{i=1}^{m}\zeta^{(i)}$$

$$\text{subject to} \quad t^{(i)}(\mathbf{w}^T \cdot \mathbf{x}^{(i)} + b) \geq 1 - \zeta^{(i)} \quad \text{and} \quad \zeta^{(i)} \geq 0 \quad \text{for } i = 1, 2, \cdots, m$$

This and the hard margin optimization problems belong to a special class of optimization problems:

"Quadratic Programming (QP) problems"

source: An Intro. to Statistical Learning, James et al

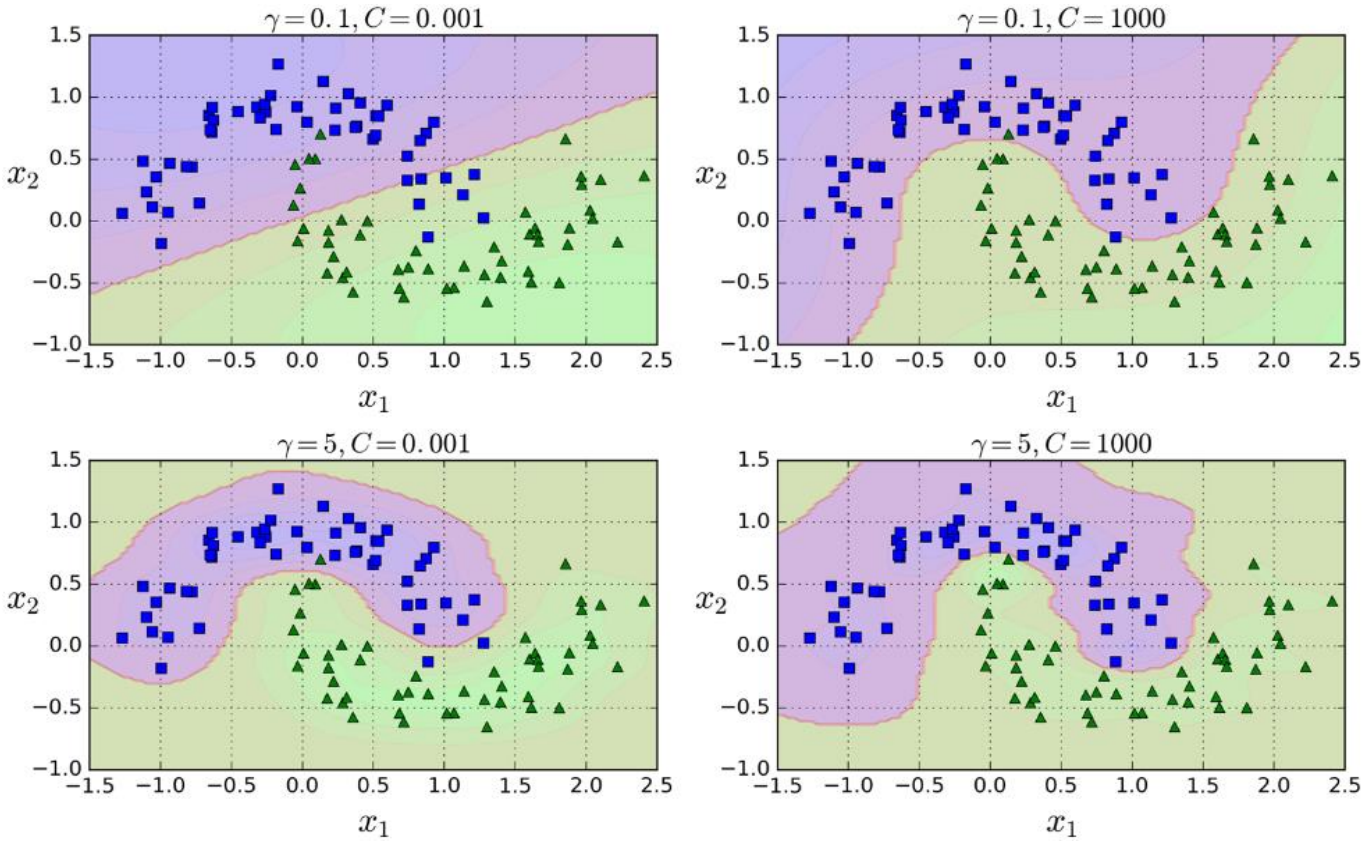
SVM: kernels

- polynomial features, similarity features
- adding the features manually
- polynomial and Gaussian RBF kernels
- regression with SVMs
- Scikit-Learn classes, and their performance

Similarity features with a kernel

```
rbf_kernel_svm_clf = Pipeline((
        ("scaler", StandardScaler()),
        ("svm_clf", SVC(kernel="rbf", gamma=5, C=0.001))
    ))
rbf_kernel_svm_clf.fit(X, y)
```

support vector classifier with a "Gaussian RBF kernel"



γ: larger value makes bell-shaped curver narrower; reduces landmark's "range of influence"

source: Geron text

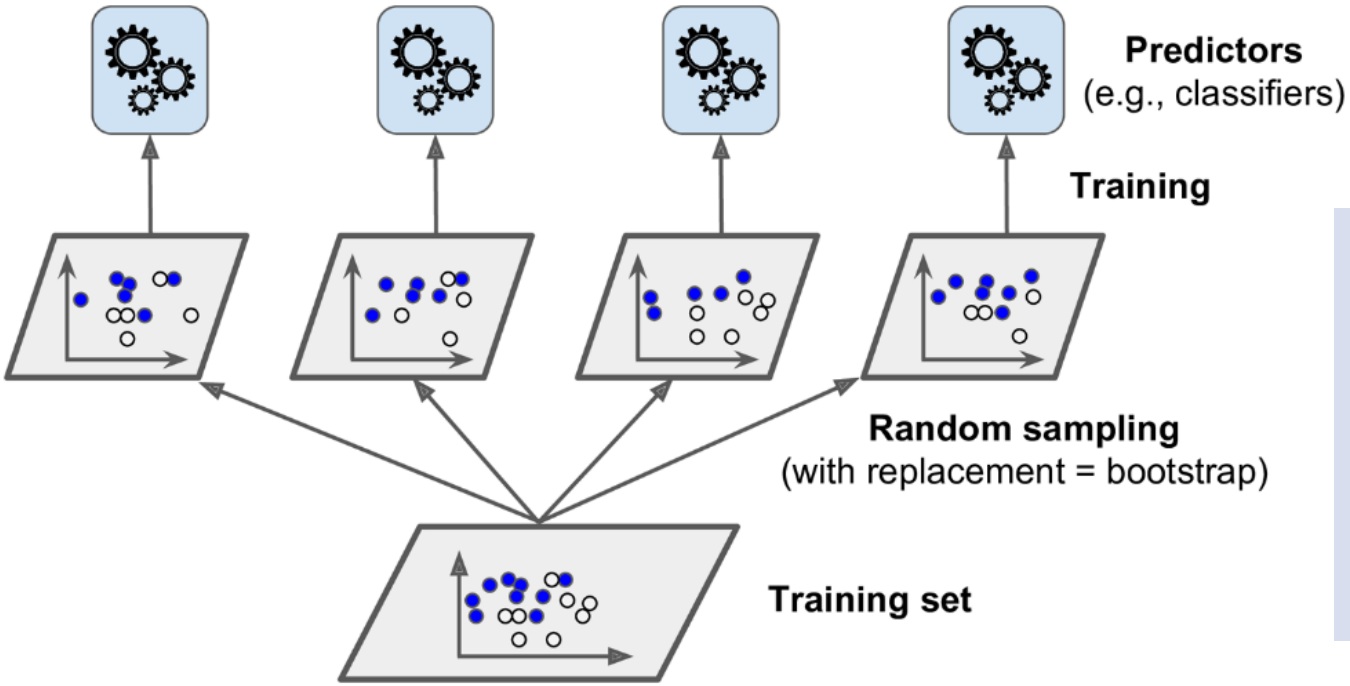
Emsemble learning: bagging

- hard and soft voting (predictor diversity)
- bagging and pasting (training data diversity)
- bagging and pasting in Scikit-Learn
- out-of-bag evaluation
- random subspaces, random patches

Bagging

In bagging, diversity of predictors is achieved by training them on different versions of the training data.

- for each predictor, train using m random samples from the training data set (with replacement)



Typically, a predictor will see about 63% of the training instances in its own training set.

Emsemble learning: AdaBoost

- random forests ~ bagging + decision trees
- AdaBoost
  - train predictors sequentially
  - training instances are weighted, based on errors of previous predictor
- AdaBoost details: predictor error rates, predictor weights, instance weights, making predictions
- AdaBoost in Scikit-Learn

AdaBoost example

training data | initial weight values

x_1	x_2	y	weight	$\hat{y}$	
.2	234	0	0.2	0	✓
.5	43	0	0.2	1	
-.1	54	1	0.2	1	✓
.6	3	0	0.2	0	✓
.3	302	1	0.2	0	

error rate is 0.4

predictor weight is 0.405

updated weight values

predictor 1

x_1	x_2	y	weight	$\hat{y}$	
.2	234	0	0.167		
.5	43	0	0.25		
-.1	54	1	0.167		
.6	3	0	0.167		
.3	302	1	0.25		

predictor 2

Emsemble learning: Gradient Boosting

- Concept of gradient boosting:
  - sequentially train predictors
  - a predictor trains on residuals from previous predictor
  - ensemble predicts sum of the base predictions
- Gradient boosting in Scikit-Learn
- Classification with gradient boosting
- Stacking: train a 'blender'

Visualization: predictors 1 and 2