10-601 Machine Learning	**Name:**
Fall 2018	**Andrew Email:**
Midterm Exam	**Room:**
10/25/2018	**Seat:**
Time Limit: 150 minutes	**Exam Number:**

Instructions:

- Fill in your name and Andrew ID above. Be sure to write neatly, or you may not receive credit for your exam.
- This exam contains 13 pages (including this cover page).
  There are 9 sections.
  The total number of points is 0.
- Clearly mark your answers in the allocated space **on the front of each page.** If needed, use the back of a page for scratch space, but you will not get credit for anything written on the back of a page. If you have made a mistake, cross out the invalid parts of your solution, and circle the ones which should be graded.
- Look over the exam first to make sure that none of the 13 pages are missing. The problems are of varying difficulty, so you may wish to pick off the easy ones first.
- No electronic devices may be used during the exam.
- Please write all answers in pen.
- You have 150 minutes to complete the exam. Good luck!

Topic	**Section**	**Points**
Decision Trees	1	11
KNN	2	4
Perceptron	3	5
Feature Engineering	4	5
Regression, Regularization, and Optimization	5	10
Error	6	4
Linear Models	7	20
Backprop for Neural Nets	8	11
Decision Boundaries	9	6
—	**Total**	0

Instructions for Specific Problem Types

For "Select One" questions, please fill in the appropriate bubble completely:

Select One: Who taught this course?

● Matt Gormley

○ Marie Curie

○ Noam Chomsky

If you need to change your answer, you may cross out the previous answer and bubble in the new answer:

Select One: Who taught this course?

● Matt Gormley

○ Marie Curie

⊗ Noam Chomsky

For "Select all that apply" questions, please fill in all appropriate squares completely:

Select all that apply: Which are scientists?

■ Stephen Hawking

■ Albert Einstein

■ Isaac Newton

□ I don't know

Again, if you need to change your answer, you may cross out the previous answer(s) and bubble in the new answer(s):

Select all that apply: Which are scientists?

■ Stephen Hawking

■ Albert Einstein

■ Isaac Newton

☒ I don't know

For questions where you must fill in a blank, please make sure your final answer is fully included in the given space. You may cross out answers or parts of answers, but the final answer must still be within the given space.

Fill in the blank: What is the course number?

10-601

10-~~7~~601

1 Decision Trees

1. **Perceptron Trees:** To exploit the desirable properties of decision tree classifiers and perceptrons, Adam came up with a new algorithm called “perceptron trees”, which combines features from both. Perceptron trees are similar to decision trees, however each leaf node is a perceptron, instead of a majority vote.

   To create a perceptron tree, the first step is to follow a regular decision tree learning algorithm (such as ID3) and perform splitting on attributes until the specified maximum depth is reached. Once maximum depth has been reached, at each leaf node, a perceptron is trained on the remaining attributes which have not been used up in that branch. Classification of a new example is done via a similar procedure. The example is first passed through the decision tree based on its attribute values. When it reaches a leaf node, the final prediction is made by running the corresponding perceptron at that node.

   Assume that you have a dataset with 6 binary attributes **(A, B, C, D, E, F)** and two output labels **(-1 and 1)**. A perceptron tree of depth 2 on this dataset is given below. Weights of the perceptron are given in the leaf nodes. Assume bias=1 for each perceptron:

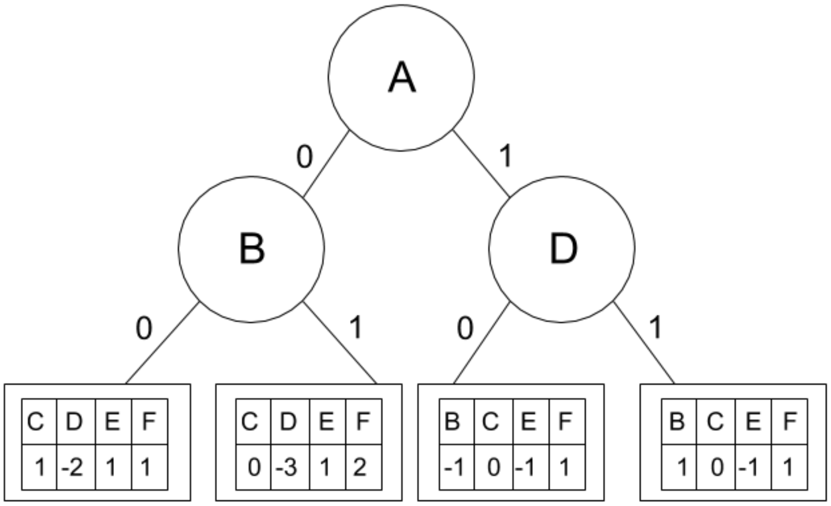


Figure 1: Perceptron Tree of max depth=2

   (a) **Numerical answer:** Given a sample $\mathbf{x} = [1, 1, 0, 1, 0, 1]$, predict the output label for this sample

   (b) **True or False:** The decision boundary of a perceptron tree will *always* be linear.

   ○ True

○ False

(c) **True or False:** For small values of max depth, decision trees are *more* likely to underfit the data than perceptron trees

○ True

○ False

2 KNN

1. **True or False:** Consider a binary (two classes) classification problem using k-nearest neighbors. We have n 1-dimensional training points $\{x_1, x_2, ..., x_n\}$ with $x_i \in \mathbb{R}$, and their corresponding labels $\{y_1, y_2, ..., y_n\}$ with $y_i \in \{0, 1\}$.

   Assume the data points $x_1, x_2, ..., x_n$ are sorted in the ascending order, we use Euclidean distance as the distance metric, and a point can be it's own neighbor. True or False: We **CAN** build a decision tree (with decisions at each node has the form "$x \geq t$" and "$x < t$", for $t \in \mathbb{R}$) that behave exactly the same as the 1-nearest neighbor classifier, on this dataset.

   ○ True

   ○ False

2. **Select all that apply:** Please select all that apply about kNN in the following options:

   Assume a point can be its own neighbor.

   □ k-NN works great with a small amount of data, but struggles when the amount of data becomes large.

   □ k-NN is sensitive to outliers; therefore, in general we decrease k to avoid overfitting.

   □ k-NN can only be applied to classification problems, but it cannot be used to solve regression problems.

   □ We can always achieve zero training error (perfect classification) with k-NN, but it may not generalize well in testing.

3 Perceptron

1. **Select all that apply:** Let $S = \{(\mathbf{x}^{(1)}, y^{(1)}), \cdots, (\mathbf{x}^{(n)}, y^{(n)})\}$ be n linearly separable points by a separator through the origin in $\mathbb{R}^d$. Let S' be generated from S as: $S' = \{(c\mathbf{x}^{(1)}, y^{(1)}), \cdots, (c\mathbf{x}^{(n)}, y^{(n)})\}$, where $c > 1$ is a constant. Suppose that we would like

to run the perceptron algorithm on both data sets separately, and that the perceptron algorithm converges on S. Which of the following statements are true?

□ The mistake bound of perceptron on S' is larger than the mistake bound on S

□ The perceptron algorithm when run on S and S' returns the same classifier, modulo constant factors (i.e., if $\mathbf{w}_S$ and $\mathbf{w}_{S'}$ are outputs of the perceptron for S and S', then $\mathbf{w}_S = c_1\mathbf{w}'_S$ for some constant c_1).

□ The perceptron algorithm converges on S'.

2. **True or False:** We know that if the samples are linearly separable, the perceptron algorithm finds a separating hyperplane in a finite number of steps. Given such a dataset with linearly separable samples, select whether the following statement is True or False: The running time of the perceptron algorithm depends on the sample size n.

○ True

○ False

4 Linear Regression

1. **Short answer:** Assume we have data $\mathbf{X} \in \mathbb{R}^{n\times d}$ with label $\mathbf{y} \in \mathbb{R}^n$. If the underlying distribution of the data is $\mathbf{y} = \mathbf{X}\beta^* + \epsilon$, where $\epsilon \sim N(0, \mathbf{I})$. Assume the closed form solution $\hat{\beta}$ for mean squared error linear regression exists for this data, write out $\hat{\beta}$'s distribution:

---

2. Consider linear regression on 1-dimensional data $\mathbf{x} \in \mathbb{R}^n$ with label $\mathbf{y} \in \mathbb{R}^n$. We apply linear regression in both directions on this data, i.e., we first fit y with x and get $y = \beta_1 x$ as the fitted line, then we fit x with y and get $x = \beta_2 y$ as the fitted line. Discuss the relations between β_1 and β_2:

(i) **True or False:** The two fitted lines are always the same, i.e. we always have $\beta_2 = \frac{1}{\beta_1}$.

○ True

○ False

(ii) **Numerical answer:** We further assume that $\mathbf{x}^T\mathbf{y} > 0$. What is the minimum value of $\frac{1}{\beta_1} + \frac{1}{\beta_2}$?

□

5 Optimization

1. **Select all that apply:** Which of the following are correct regarding Gradient Descent (GD). Assume data log-likelihood is $L(\theta|X)$, which is a function of the parameter θ, and the objective function is negative log-likelihood .

   □ GD requires that $L(\theta|X)$ is concave with respect to parameter θ in order to converge

   □ GD requires that $L(\theta|X)$ is convex with respect to parameter θ in order to converge

   □ GD update rule is $\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta|X)$

   □ Given a fixed small learning rate (say $\alpha = 10^{-10}$), GD will always reach the optimum after infinite iterations (assume that the objective function satisfies the convergence condition).

2. **Select all that apply:** Which of the following are correct regarding Gradient Descent (GD) and stochastic gradient descent (SGD)

   □ Each update step in SGD pushes the parameter vector closer to the parameter vector that minimizes the objective function.

   □ The gradient computed in SGD is, in expectation, equal to the gradient computed in GD.

   □ The gradient computed in GD has a higher variance than that computed in SGD, which is why in practice SGD converges faster in time than GD.

3. Let $X_1, X_2, ..., X_N$ be i.i.d. data from a uniform distribution over a diamond-shaped area with edge length $\sqrt{2}\theta$ in $\mathbb{R}^2$, where $\theta \in \mathbb{R}^+$ (see Figure 2). Thus, $X_i \in \mathbb{R}^2$ and the distribution is

$$p(x|\theta) = \begin{cases} \frac{1}{2\theta^2} & if\ \|x\| \leq \theta \\ 0 & otherwise \end{cases}$$

   where $\|x\| = |x_1| + |x_2|$ is $L1$ norm. Please find the maximum likelihood estimator of θ.

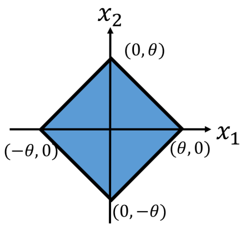


Figure 2: Area of $\|x\| \leq \theta$

4. **Short answer:** Suppose we want to model a 1-dimensional dataset of N real valued

features $\left(x^{(i)}\right)$ and targets $\left(y^{(i)}\right)$ by:

$$y^{(i)} \sim \mathcal{N}\left(\exp(wx^{(i)}), 1\right)$$

Where w is our unknown (scalar) parameter and $\mathcal{N}$ is the normal distribution with probability density function:

$$f(a)_{\mathcal{N}(\mu,\sigma^2)} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right)$$

Can the maximum conditional negative log likelihood estimator of w be solved analytically? If so, find the expression for w_{MLE}. If not, say so and write down the update rule for w in gradient descent.

6 Logistic Regression

1. We have a set of 2-dimensional data points $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(n)}\}$ with $\mathbf{x}^{(i)} \in \mathbb{R}^2$, and their corresponding labels $\{y_1, y_2, ..., y_n\}$ with $y_i \in \{0, 1\}$ shown below in figure 3. Let "○" denote 1 and "×" denote 0.

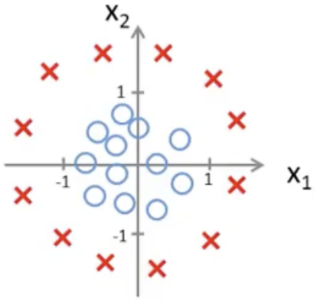


Figure 3: Non-linear classification boundary

(i) **True or False:** We can engineer a set of features such that a binary logistic regression model trained on the transformed features perfectly categorizes this dataset. Notice, the unmodified feature space is $\mathbf{x}^{(i)} = [1, x_1, x_2]^T$ after folding in the bias term, as introduced in class.

○ True

○ False

(ii) If true, please provide the engineered feature space using which a binary logistic regression can perfectly categorize the above dataset. If false, please explain why.

2. By appropriately transforming features, binary logistic regression model can solve non-linear classification problems. Can we learn the decision boundary in figure 6 by transforming features x_1 and x_2? If no, why not? If yes, what features are sufficient? (Define symbols "+" and "-" as label 1 and label 0, respectively. $x = [x_1, x_2]^T \in \mathbb{R}^2, y \in \{0, 1\}, \theta_i \in \mathbb{R}$.)

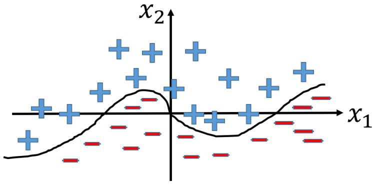


Figure 4: Non-linear classification boundary

○ Yes

○ No

3. True/ False:

   - The logistic negative log-likelihood has a global minimum
   - the logistic negative log-likelihood has a closed-form solution
   - The log-loss can continue decreasing even after the model achieves perfect training accuracy

4. Let $D = (\mathbf{x}^{(i)}, y^{(i)})_{i=1}^n$ be a nonempty probabilistic classification training dataset, $\mathbf{x}^{(i)} \in \mathbb{R}^p$ and $y^{(i)} \in \{0, 1\}$ for all $i \in \{1, .., n\}$. Let $\mathbf{w} \in \mathbb{R}^p$ be a binary logistic regression weight vector(without an explicit bias term) **that would assign the correct labels to all examples in** D. Respond to the following statement with "true" or "false" followed by some justification:

   There is a $p \times p$ dimensional matrix $\mathbf{A}$ depending only on p (not $\mathbf{w}$ nor the training set) such that $\mathbf{Aw}$ is guaranteed to assign a strictly greater likelihood to the training data than $\mathbf{w}$ does.

5. **Select one:** In binary logistic regression, given parameters θ, we use the function $f(x) = \frac{1}{1+e^{-\theta^T x}}$ to perform classification. We predict x's label as 1 if $f(x) \geq t$ and as 0 if $f(x) < t$, where $t = 0.5$ for this particular choice of $f(x)$. The choice of $f(x)$ is up to us (recall in lecture notes we started with non-differentiable $sign(\theta^T x)$). Now consider another suitable function $g(x) = \frac{e^{\theta^T x} - e^{-\theta^T x}}{e^{\theta^T x} + e^{-\theta^T x}}$ with same θ for binary classification, which of the following statements is true?

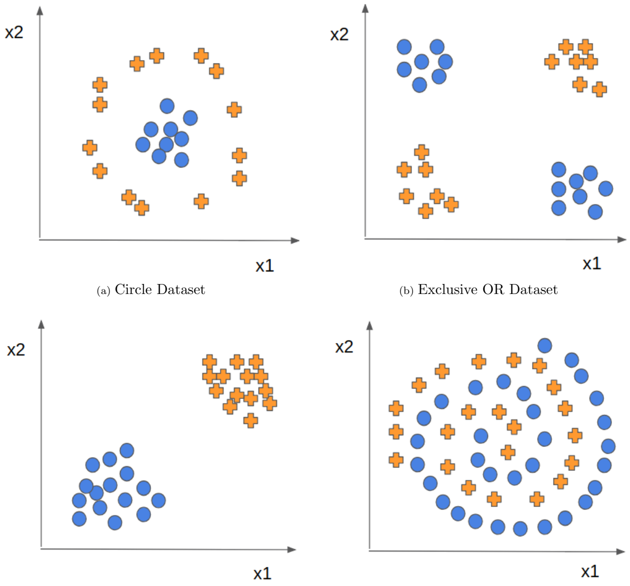
   ○ $f(x)$ and $g(x)$ classify x similarly, $t = 0.5$ for $g(x)$

   ○ $f(x)$ and $g(x)$ classify x differently, $t = 0.5$ for $g(x)$

   ○ $f(x)$ and $g(x)$ classify x similarly, $t = 0$ for $g(x)$

   ○ $f(x)$ and $g(x)$ classify x differently, $t = 0$ for $g(x)$

6. **Select one:** On which of the following datasets can a logistic regression classifier $p(y = 1|x, \theta) = \frac{1}{1+e^{-\theta^T x}}$, $x = [x_1, x_2]^T$ achieve zero training error?



(a) Circle Dataset (b) Exclusive OR Dataset

(c) Gaussian Dataset (d) Spiral Dataset

○ Circle dataset

○ Exclusive OR dataset

○ Gaussian dataset

○ Spiral dataset

7 Regularization / Feature Engineering

1. **Pick the true statement:** Recall in regularization, given an objective function $J(\Theta)$, we have

$$\hat{\Theta} = \arg\min_{\Theta} J(\Theta) + \lambda r(\Theta)$$

where $r(\Theta)$ is the regularization term.

□ The recommended way to choose λ is to pick the value of λ that minimizes the **training set error**.

□ The recommended way to choose λ is to pick the value of λ that minimizes the **cross validation error**.

□ The recommended way to choose λ is to pick the value of λ that minimizes the **test set error**.

8 Neural Networks

1. **Select all that apply:** The XOR function for two binary variables x and y is defined as follows:

$$\begin{aligned} f(x, y) &= 0, x = y \\ &= 1, x \neq y \end{aligned}$$

Based on this information, select all statements which are true:

□ The XOR function is linearly separable

□ A single-layer perceptron is incapable of representing the XOR function

□ A multi-layer perceptron is incapable of representing the XOR function

□ None of the above

9 Hidden Markov Models

1. **Select all that apply:** Let there be an HMM with timesteps = T. Let b be the emission matrix and π the vector of priors. $b_j(x_i) = P(x_i|y = j)$. α and β have the usual definition in the forward backward algorithm. The observations are denoted by x_1, ...x_T. Let J be the total number of states. Which of the following are equal to $P(x_{1:T})$-

□ $\sum_{j=1}^{J} \beta_1(j)$

□ $\sum_{j=1}^{J} \alpha_T(j)$

□ $\sum_{j=1}^{J} \beta_1(j)\pi_j b_j(x_1)$

□ $\sum_{j=1}^{J} \alpha_t(j)\beta_t(j)$

2. In the HMM assignment, we used the normalization trick to avoid underflow. However, after normalization, we are not computing the correct likelihood anymore. Formally, we use M to denote the number of states, T to denote the length of sequence. We use x to denote the observation sequence and y the hidden state sequence. In each forward step, before computing α_{t+1}, we first normalize α_t such that $\sum_{j=1}^{M} \alpha_t(j) = 1$. We do not perform normalization at α_T. In this case, if we still use $\sum_{j=1}^{M} \alpha_T(j)$ to compute the likelihood, then instead of $P(x_{1:T})$, what quantity are we actually computing? (Write in terms of probabilities, with x, y, M and T.)

3. In order to avoid the problem discussed in the previous question, briefly (in one sentence) describe how you will compute the correct log likelihood while still avoiding underflow:

4. Consider the following dataset:
   black_A car_N
   black_A jeep_N
   jeep_N is_V black_N
   where for each a_b, a is a word and b is a tag.

   For the set of tags = {A, N, V} and words = {black, car, jeep, is}, draw a finite state diagram representing state transitions for this dataset.
   Note: Please use the same counting strategy used on the assignment to perform MLE

10 Reinforcement Learning

1. Let revisit the maze world we have seen in assignment 8. The rules are defined as such:

State 1	G
State 2	H
State 3	State 4

Table 1: Map of the maze

- At each valid state (State 1-4), the agent can move North, South, West, and East.
- **Stochastic Environment**: After the agent selects an action, it has 50% chance entering its target state, 25% chance to the left (of the target state), and 25% chance entering the state to the right (of the target state). For example, If an agent is at state 3, and decides to go right. It has 50% chance ending up at state 4, 25% going to state 2, and 25% staying in its previous state (hit the wall by going south).
- If the agent hits a wall (edge of the maze) or an obstacle (H), it stays in its previous state.

- The agent receives a reward of 100 when entering the goal state (G), 0 otherwise.
- Discount Factor = 1

Please perform two rounds of value iteration, and report the state value function (V) after each round. The initial state values (including G) are 0.

Fill in the tables:

0	0
0	H
0	0

Table 2: Initial state

	H

Table 3: Round 1

	H

Table 4: Round 2

2. **Short answer:** For MDPs, recall the value function of a state s under a policy π, denoted $v_\pi(s)$, is the expected discounted return when starting in s and following π thereafter:

$$v_\pi(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right]$$

Similarly, the value of taking action a in state s under a policy π, denoted $q_\pi(s,a)$, is the expected discounted return starting from s, taking the action a, and thereafter following policy π:

$$q_\pi(s,a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right]$$

Finally note that if the agent is following policy π at time step t, then $\pi(a \mid s)$ is the probability that $A_t = a$ if $S_t = s$, that is we take action a at time step t given we are at state s with probability $\pi(a \mid s)$. This allows our policy to be stochastic.

Write down the equation for $v_\pi(s)$ in terms of $q_\pi(s,a)$ and $\pi(a \mid s)$:

11 SVM, PCA

1. **Select all that apply:** In class, we learned that SVM decision boundary is determined and affected by support vectors. Now, let's define support vectors to be the data points that are left behind, as you continue removing data points one by one, such that the original decision boundary remains unchanged. In 2-D binary linearly separable classification case, how many support vectors may possibly be left behind to make the original decision boundary unchanged?

□ 1

□ 2

□ 3

□ 4

□ more than 4

2. **Select all that apply:** Which of the following decision boundary can be generated by using the kernel $K(u, v) = u^T v + (u^T v)^2 + c$?

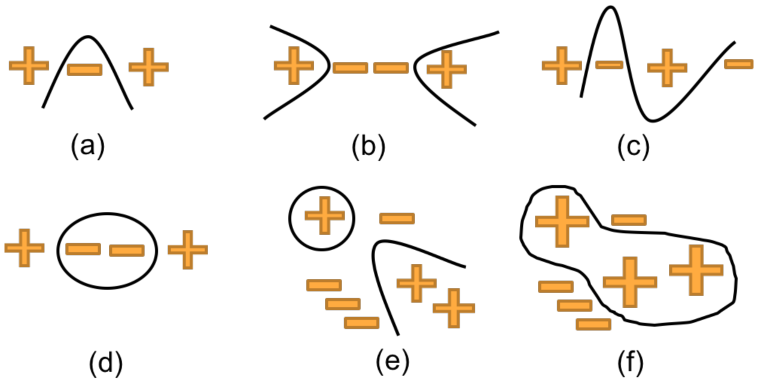


Figure 6: Decision boundary

□ a

□ b

□ c

□ d

□ e

□ f

3. **Select One:** Consider N points $\{\mathbf{x}^{(i)}\}_{i=1}^N$ in $\mathbb{R}^d$, you pick p principal components after applying PCA on these points. Can you always reconstruct any point $\mathbf{x}^{(i)}$ for all $i \in \{1..N\}$ from these components with zero reconstruction error?

○ if $p < N$, yes

○ if $p < d$, yes

○ if $p = d$, yes

○ no always