Data C100/C200, Midterm

Fall 2023

Name: ______________________________

Email: ______________________________@berkeley.edu

Student ID: ______________________________

Name and SID of the person on your right: ______________________________

Name and SID of the person on your left: ______________________________

Instructions:

This midterm exam consists of **73 points** spread out over **7 questions** and the Honor Code and must be completed in the **110 minute** time period, unless you have accommodations supported by a DSP letter.

Note that some questions have circular bubbles to select a choice. This means that you should only **select one choice**. Other questions have boxes. This means you should **select all that apply**. Please shade in the box/circle to mark your answer.

You must write your Student ID number at the top of each page.

Points Breakdown:

Question	Q1	Q2	Q3	Q4	Q5	Q6	Q7
Points	18	8	6	15	9	8	8

Honor Code [1 Pts]:

As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others. I am the person whose name is on the exam and I completed this exam in accordance with the Honor Code.

Signature: ______________________________

1 Pound It, Noggin [18 Pts]

The YouTube channel Dude Perfect is known for its competition videos. In these multi-round competitions, members either compete individually or in teams of 2. Data collected from these videos is stored in a `DataFrame` called `winners`. Assume that each video only had one winning person/team. The first few rows (as well as descriptions of each column) are provided below.

- `Title`: Title of the video.
- `Rounds`: The number of rounds of competition that took place in the video.
- `Duration`: Duration of the video in minutes.
- `Views`: Millions of views the video received.
- `Name`: Name of the person or team who won.
- `Team Contest`: Binary column indicating 1 if it was a team competition, 0 if it was an individual competition.

	Title	Rounds	Duration	Views	Name	Team Contest
0	Angry Birds in Real Life	4	10.971	5.4	Garrett	0
1	The Survivor Games	6	21.422	7.5	Team Bluegrass	1
2	$50,000 Crystal Treasure Hunt	1	13.974	6.6	Cory	0
3	World's Strongest Dude	4	13.853	16.0	Ty	0
4	Ultimate Mini Games Battle 3	13	12.522	16.0	Ty	0
5	Deep Sea Fishing Battle 2	1	13.658	8.3	Nacho Boat	1

(a) [1 Pt] What is the **granularity** of `winners`? Respond in one sentence.

(b) [2 Pts] Which **variable type** best describes each of the following columns of `winners`?

	Quantitative Continuous	Quantitative Discrete	Qualitative Ordinal	Qualitative Nominal
(i) `"Rounds"`	○	○	○	○
(ii) `"Duration"`	○	○	○	○
(iii) `"Name"`	○	○	○	○
(iv) `"Team Contest"`	○	○	○	○

The remainder of this question involves coding. All code for each part, where applicable, must be written in Python. Assume that the `pandas` library has been imported as `pd` and the `numpy` library has been imported as `np`.

(c) [3 Pts] Cory, Coby, and Cody are all members of Dude Perfect. Select all the lines of code below which correctly outputs a `DataFrame` with the columns `Title` and `Name`, where the first 2 letters of `Name` are `"Co"`.

☐
```
winners[winners['Name'].str.split('Co') == True]
.iloc[:, [0, 4]]
```

☐
```
winners[winners['Name'].str[0:2] == 'Co']
[['Title', 'Name']]
```

☐
```
winners[winners['Name'].str.contains(r'(^Co)', regex=True)]
.loc[:, ['Title', 'Name']]
```

☐
```
winners[winners['Name'].str.contains('Co')]
.loc[['Title', 'Name']]
```

☐
```
winners[winners['Name'].str == 'Co'][['Title', 'Name']]
```

☐
```
winners[winners['Name'].str[:2].isin(['Co'])]
.iloc[['Title', 'Name']]
```

(d) [2 Pts] Select all of the following lines of code that correctly return a `Series` containing the winner that appears the most number of times in the `DataFrame`. You may assume that there is no tie for the most number of appearances. The result should be a `Series` with a length of one, containing the name of the person and the corresponding number of times they've won.

☐
```
winners.groupby('Name').count()
.sort_values('count', ascending = False)['count'][0]
```

☐
```
winners.groupby('Name').size()
.sort_values(ascending = False).head(1)
```

☐
```
winners.groupby('Name').size().sort_values().head(1)
```

☐
```
winners['Name'].value_counts()[0:1]
```

(e) [4 Pts] Sean wants to know which members/teams of Dude Perfect average more than 6 million `Views` for contests that they win. Help him fill in the blanks of the following lines of code to achieve this. The output should have the same structure as `winners` but only contains rows where `Name` belongs to a person or team averaging more than 6 million views.

```
winners.____A____(____B____).____C____(lambda sf: __________D__________)
```

(i) Fill in blank `A`:

(ii) Fill in blank `B`:

(iii) Fill in blank `C`:

(iv) Fill in blank `D`:

(f) [3 Pts] The first five rows of a new `DataFrame` called `teams` is given to you below.

	Team	First Member	Second Member
0	Team Bluegrass	Cody	Coby
1	Team Redwood	Garrett	Sparky
2	Yellow Jackets	Ty	Cory
3	Wet Willies	Ty	Sparky
4	Nacho Boat	Garrett	Cory

Fill in the blanks to create a `DataFrame` called `combined`, which includes information about the winning team for each team contest, and the video they won. Your resulting `DataFrame` should have the exact same columns as the example given below. You may assume team names were never repeated between videos.

Hint: Remember to remove columns that are not included in `combined` given below.

```
combined = winners.___A___(___B___,______C_______,_______D_______)
                  .______________E______________
combined.head(2)
```

	Title	Rounds	Duration	Views	Team	First Member	Second Member
0	The Survivor Games	6	21.422	7.5	Team Bluegrass	Cody	Coby
1	Deep Sea Fishing Battle 2	1	13.658	8.3	Nacho Boat	Garrett	Cory

(i) Fill in blank A:

(ii) Fill in blank B:

(iii) Fill in blank C:

(iv) Fill in blank D:

(v) Fill in blank E:

(g) [3 Pts] Now that we know the members of each team, use `combined` to write a line of code that displays the median amount of views that each pairing of Dude Perfect members received for contests that they won. Two team members can team up multiple times across videos, but each team name will always be unique across all videos.

You may only use one function in your answer. **If a pair has never teamed up to win together, they should have a value of 0**.

2 pan-DONS [8 Pts]

Welcome back to Day 100 of Guessing Brandon's lunch! Every day, members of the course staff take one guess each for what Brandon is going to eat for lunch. These predictions are stored in `guesses`, the first five rows of which are given to you below.

	Date	Name	Guess	Correct
0	10/18/2023	Matthew	Taco Sinaloa	0
1	10/18/2023	Yuerou	Toss Noodle	1
2	10/18/2023	Manas	Thai Basil	0
3	10/18/2023	Pragnay	Mezzo	0
4	10/17/2023	Manas	Bongo Burger	-99

Each day, Brandon goes through and manually fills in the `Correct` column with a value of `1` if the `Guess` was correct and `0` if the `Guess` was incorrect. However, every 3-4 days, he forgets what he had for lunch, and gives every guesser on that day a value of `-99` to represent a missing value (assume these values make up around 30% of the `Guess` column).

(a) [1 Pt] Which of the following is the **BEST** way we can deal with missing values (`-99`)?

◯ Dropping all rows with -99 values

◯ Imputing with 0

◯ Imputing with 0.5

◯ Imputing with the average of that person's `Correct` column

(b) [3 Pts] Select all of the following lines of code that correctly output an integer of how many missing values (`-99`) are in the column `Correct`.

☐ `guesses[guesses['Correct'] < 0]['Correct']`

☐ `sum(guesses['Correct'])-len(guesses[guesses['Correct']<0])`

☐ `guesses[guesses['Correct'] == -99]['Correct'].count()`

☐ `(guesses['Correct'] == -99).sum()`

☐ `sum(guesses[guesses['Correct']<0]['Correct'].value_counts())`

☐ `guesses['Correct'].value_counts().loc[-99]`

(c) [4 Pts] Brandon is able to recall what he ate for lunch every day, and creates a new `DataFrame` called `guesses_clean`, which has the same format as `guesses`, but with the correct value of `0` or `1` in place of `-99` in the `Correct` column. With this new information, Brandon wants to see a summary of how course staff members performed in this contest.

Fill in the blanks to generate a `DataFrame` which contains each guesser's `Name` as the index, a column labeled `Guess` with the **total** number of guesses each person submitted and a column labeled `Correct` with the number of **correct** guesses per person.

```
guesses_clean.____A____(____B____).____C____(____________D____________)
```

(i) Fill in blank `A`:

(ii) Fill in blank `B`:

(iii) Fill in blank `C`:

(iv) Fill in blank `D`:

3 Tyrannosaurus Re(ge)x [6 Pts]

In this question, the Python regular expression library has been imported as `re`. For all parts, you will only need to worry about the example strings given to you and may assume that these examples cover all edge cases.

Congratulations to Mir on starting his new job at Triassic Park, an amusement park that brought dinosaurs back to life! His first task is to clean up some of the data records stored for the park's dinosaurs. Two examples of these records are shown in the variables `rex_string` and `tri_string` below:

```
rex_string = 'species=tyrannosaurusrex,name=Rexy,id=5884nngh'
tri_string = 'species=triceratops,name=Pointyboyo,id=8431nrfj'
```

(a) [2 Pts] As part of this data cleaning, a RegEx pattern called `dino_pattern` is created to isolate each of the different fields from the above strings using `re.findall`. Example outputs are shown below:

```
re.findall(dino_pattern, rex_string)
```

```
['species=tyrannosaurusrex', 'name=Rexy', 'id=5884nngh']
```

```
re.findall(dino_pattern, tri_string)
```

```
['species=triceratops', 'name=Pointyboyo', 'id=8431nrfj']
```

Which of the following could be `dino_pattern`?

- ○ `r',?(.*)'`
- ○ `r',?([a-z]+=\w*)'`
- ○ `r',(\w*=\S+)'`
- ○ `r'[species|name|id]=\w+'`

(b) [2 Pts] Mir's next assignment is to isolate keywords from some of the park's online reviews. He decides the best way to do this is to use a RegEx pattern, and writes the following lines of code to extract data from a string called `review_1`:

```
review_1 = "10/10! I loved my experience. We visited the T-Rex exhibit 3 times!!! 5 stars!"
re.findall(r'(\d+\s{1}\S+)!', review_1)
```

What will be the output of Mir's code? Recall that `re.findall` returns a list of matches. Format your answer like: `['string_1', 'string_2', 'string_3']`

(c) [2 Pts] A dangerous incident occurred at the park, leading to some negative reviews. Mir is tasked with a particularly negative review called `review_2` which is given below:

```
review_2 = "Truly awful awful awful. All the dinosaurs escaped their enclosures."
```

Which of the following patterns would correctly capture the substring `"awful awful awful"` in `review_2`? **Select all that apply**.

- ☐ `r'(awful\s){3}'`
- ☐ `r'awful\s\w{5}\s[a-z]*'`
- ☐ `r'\w{5}\s[a-z]+\sawful'`
- ☐ `r'Truly\s([awful]{3})'`

4 Cruel Sampler [15 Pts]

Superstar pop sensation Saylor Twift went on tour this summer, and tickets were in high demand. Fans who attended the tour either bought a ticket directly from the artist or bought their ticket on the resale market. For the purposes of this question, we'll refer to these groups as "direct" and "resale", respectively.

(a) [1 Pt] With the tour concluded, Lillian wants to know whether UC Berkeley students who bought tickets were more likely to get them direct or resale. She decided to conduct a survey to see what percentage of students fall within each group. What is the target population of this survey?

◯ All UC Berkeley students

◯ All people who bought a ticket

◯ All UC Berkeley students who bought a ticket

◯ All UC Berkeley students who bought a ticket resale

(b) [2 Pts] Suppose Lillian conducts her study by posting a link to a survey on the Data 100 Ed, and closes the link after the first 100 responses are recorded. Is her sampling frame the same as her target population?

For full credit, **please explain your reasoning using 1 sentence**. You will receive partial credit for a correct answer without giving a correct explanation.

◯ Yes ◯ No

(c) [2 Pts] Which of the following sampling procedures, errors, and biases can be used to explain Lillian's process? **Select all that apply.**

☐ Simple random sample ☐ Probability sample

☐ Convenience sample ☐ Non-response bias

Lillian aggregates the results of her survey based on the date of each show. She decides to store this data in a `DataFrame` called `tickets`, which contains the average price for `floor` seats, `lower_bowl` seats, `upper_bowl` seats, and the overall average price of `all` seats for each show. The first five rows of `tickets` are shown below.

	date	city	floor	lower_bowl	upper_bowl	all
0	03/17	Glendale	849.00	306.00	95.67	396.73
1	03/18	Glendale	259.00	277.67	111.22	182.88
2	03/31	Arlington	899.00	419.00	115.00	400.71
3	04/01	Arlington	602.67	336.00	156.50	312.92
4	04/02	Arlington	572.25	299.00	223.20	358.50

(d) [3 Pts] Lillian wants to see whether concerts with higher average prices for `floor` tickets also had higher average prices for `upper_bowl` tickets. Which of the following visualization types and function calls would plot an appropriate visualization for this scenario? **Select all that apply.**

☐ Line plot

☐ 2D KDE plot

☐ Side-by-side bar chart

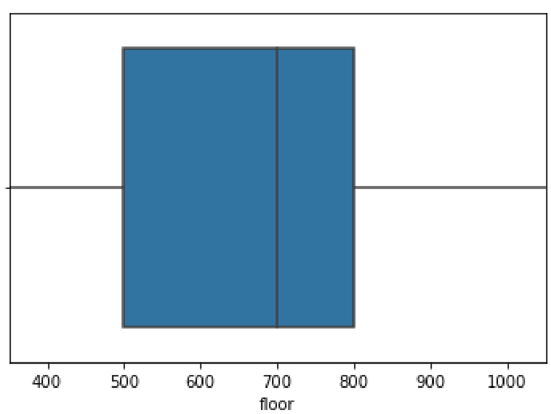
☐ Scatterplot

☐ Hexplot

☐ Overlaid histogram

(e) [3 Pts] For the following subparts, you do not need to solve or simplify any expression you write. If there is not enough information to answer a certain subpart, write "Not enough information".

Below is a boxplot generated from the `floor` column, but due to a bug in Lillian's code, the whiskers are cut off.



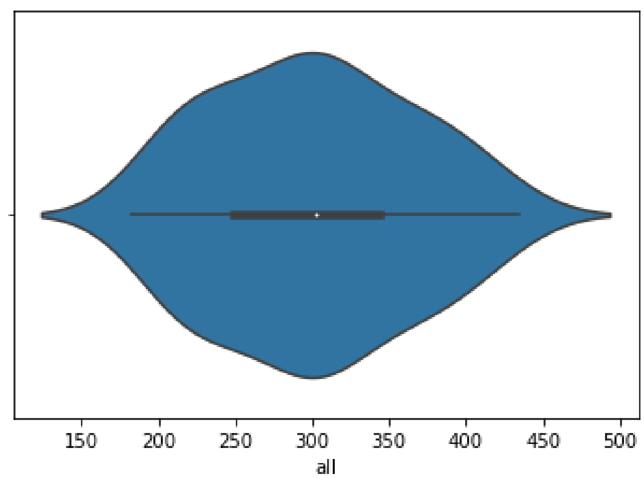
(i) Write a mathematical expression that evaluates to the IQR of the boxplot.

(ii) Write a mathematical expression that evaluates to the value of the left whisker of the boxplot.

Hint: The whiskers typically denote the boundary for what is considered an outlier. Due to a bug in Lillian's code, they do not appear in the image above.

(iii) Write a mathematical expression that evaluates to the mean of the boxplot.

(f) [2 Pts] Below is a violin plot generated from the `all` column.



Which of the following statements are **true** for this distribution? **Select all that apply**.

☐ The distribution is roughly symmetric

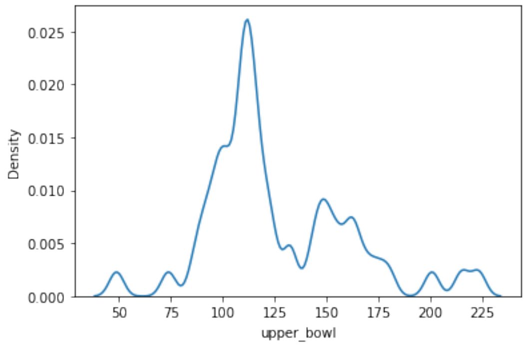
☐ The distribution is left-skewed

☐ The mode is around 300

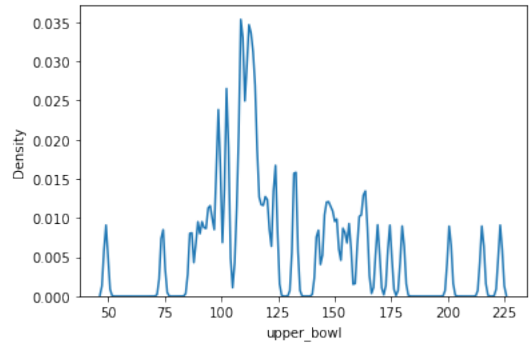
☐ The mean, median, and mode are all approximately the same value

(g) [1 Pt] For the KDEs below, list them in order of highest to lowest bandwidth parameter α.

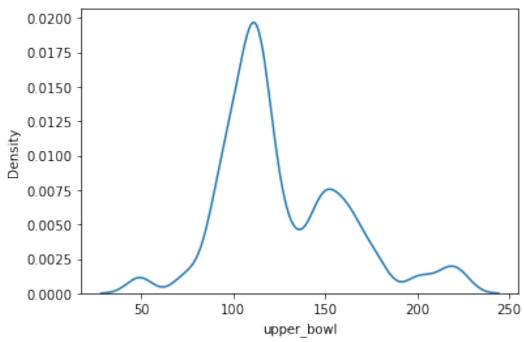
A.



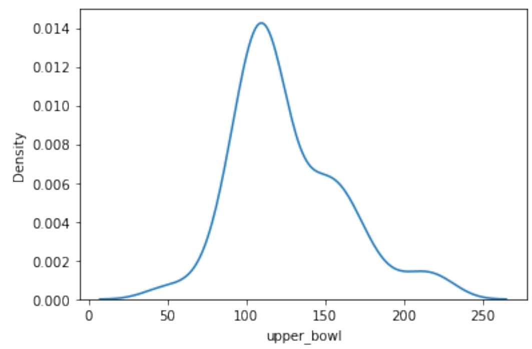
B.



C.



D.



(h) [1 Pt] Which of the following is **true** of the KDE generation process?

- ◯ Before scaling, the area under each kernel is $\frac{1}{n}$, where n is the number of data points.
- ◯ Each kernel is roughly centered at the mode of the dataset.
- ◯ In a Gaussian kernel, the bandwidth parameter represents the standard deviation of the Gaussian.
- ◯ There is always a true optimal value of α.

5 Regression Session [9 Pts]

Rohan has a set of data points $x_i \in R$ and $y_i \in R$ for $i \in \{1, ..., n\}$, and wants to use regression on inputs x_i to predict y_i values. He decides to use the following model:

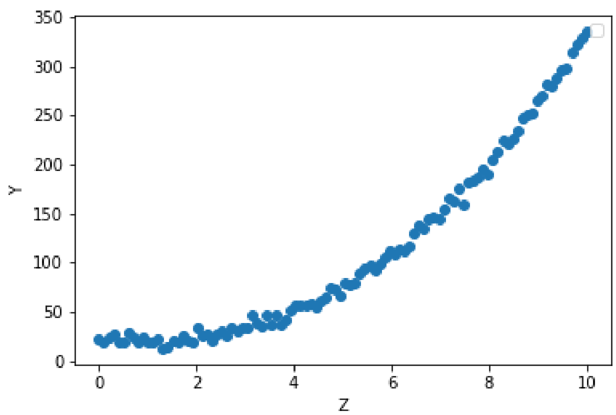
$$\hat{y}_i = \theta ln(x_i^2) \text{ with } \theta \in R$$

(a) [4 Pts] Calculate the value of $\hat{\theta}$, which minimizes the Mean Squared Error (MSE) of Rohan's model. This loss is a convex function. Please simplify and leave your answer in terms of x_i, y_i and n.

(b) [3 Pts] Which of the following functions are appropriate loss functions that also penalize outliers more harshly than MSE? **Select all that apply**.

- ☐ $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})$
- ☐ $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^4$
- ☐ $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^3$
- ☐ $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^5$
- ☐ $\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}|^3$
- ☐ $\frac{1}{n}\sum_{i=1}^{n}|(y_i - \hat{y})^5|$

(c) [2 Pts] Suppose Rohan brings in a new variable z_i to predict y_i for $i \in \{1, ..., n\}$. A scatterplot showing the relationship between z_i and y_i values is shown below.



Select the transformation which **best** forms a linear relationship between the two variables. Then in the box below, derive a relationship that expresses the **original, untransformed variable** y_i **as a function of the variable** z_i. Your answer should take the form $y_i = f(z_i)$, where f is some function of z_i that you define.

- ○ $y_i^3 = z_i$
- ○ $log(y_i) = \sqrt{z_i}$
- ○ $ln(y_i) = ln(z_i)$
- ○ $\sqrt{y_i} = z_i$

6 OLS Well That Ends Well [8 Pts]

(a) [2 Pts] Tina wants to use OLS to predict a person's height, stored in a vector $\mathbb{Y}$. She makes use of an $m \times n$ design matrix $\mathbb{X}$, which contains m rows and n columns, including a column representing intercept. Assume that $\mathbb{X}$ and $\mathbb{Y}$ meet all the requirements for OLS.

(i) How many features are in Tina's original data set?

(ii) What are the dimensions of $\mathbb{Y}$? Format your answer like $a \times b$, where a is the number of rows, and b is the number of columns.

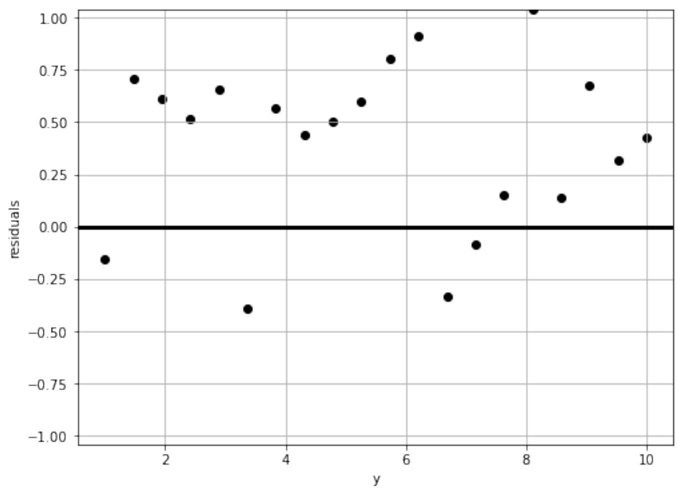
(b) [2 Pts] Which of the following statements are **true** regarding Tina's model? **Select all that apply**.

- ☐ $\mathbb{X}$ must be invertible to achieve a unique solution.
- ☐ Tina can use either MSE or MAE as her loss function.
- ☐ The parameter vector $\hat{\theta}$ minimizes the sum of the residuals ($\mathbb{Y} - \hat{\mathbb{Y}}$).
- ☐ The length of the parameter vector $\hat{\theta}$ is equal to n.

(c) [2 Pts] Columns in $\mathbb{X}$ represent data such as a person's weight and their parents' height. Which of the following subsets of columns would yield a unique solution for OLS ***and*** have the residuals sum up to 0? **Select all that apply**

- ☐ Parent 1's height (inches), Parent 2's height (inches), Person's weight
- ☐ Parent 1's height (inches), Parent 2's height (inches), Parent 2's height (meters), intercept column
- ☐ Parent 1's height (inches), Parent 2's height (meters), Person's weight, intercept column
- ☐ Parent 1's height (inches), Parent 2's height (inches), intercept column

(d) [2 Pts] This subpart is unrelated to the previous subparts. Below is a plot of residuals.



Based on this graph, which of the following statements must be **true**? **Select all that apply**.

☐ The model is consistently underpredicting.

☐ The design matrix is not full column rank.

☐ One or more of the columns used to predict $\hat{y}$ must undergo a log transformation.

☐ The model does not include an intercept term.

7 Descent Delight [8 Pts]

(a) [2 Pts] Which of the following statements are **true**? **Select all that apply**.

☐ For one iteration, stochastic gradient descent is slower than batch gradient descent.

☐ If the current gradient is negative, α is updated to be negative.

☐ The choice of starting point matters for gradient descent.

☐ Learning rate does not impact whether or not gradient descent converges, only how long it would take to do so.

(b) [6 Pts] Shiny wants to learn more about the different types of gradient descent. For each of the following sub-parts, determine which types of gradient descent could match the description. **Select all that apply**.

(i) At each iteration, the gradient is approximated and may not descend towards the true minimum.

☐ Batch gradient descent

☐ Mini-batch gradient descent

☐ Stochastic gradient descent

☐ None of the above

(ii) Each update of θ may jump back and forth between two sides of the optimal $\hat{\theta}$.

☐ Batch gradient descent

☐ Mini-batch gradient descent

☐ Stochastic gradient descent

☐ None of the above

(iii) Gradient descent is guaranteed to converge to the global minima for convex functions for **all** values of α.

☐ Batch gradient descent

☐ Mini-batch gradient descent

☐ Stochastic gradient descent

☐ None of the above

You are done with the Midterm! Congratulations!

- **Make sure that you have written your student ID number on *every page* of the exam.** You may lose points on pages where you have not done so
- Also ensure that you have **signed the Honor Code** on the cover page of the exam for 1 point.

The following questions are worth no points and are just for fun!

Which pet do you think should have won the Data 100 Cutest Pet Contest from Homework 5?

◯ Appa

◯ Pishi

◯ Mimi

Use this box to draw your favorite Data 100 moment or experience so far!

This page has been intentionally left blank.

Fall 2023 Data C100/C200 Midterm Reference Sheet

Pandas

Suppose `df` is a DataFrame; `s` is a Series. `import pandas as pd`

Function	Description
`df[col]`	Returns the column labeled `col` from `df` as a Series.
`df[[col1, col2]]`	Returns a DataFrame containing the columns labeled `col1` and `col2`.
`s.loc[rows] / df.loc[rows, cols]`	Returns a Series/DataFrame with rows (and columns) selected by their index values.
`s.iloc[rows] / df.iloc[rows, cols]`	Returns a Series/DataFrame with rows (and columns) selected by their positions.
`s.isnull() / df.isnull()`	Returns boolean Series/DataFrame identifying missing values
`s.fillna(value) / df.fillna(value)`	Returns a Series/DataFrame where missing values are replaced by `value`
`s.isin(values) / df.isin(values)`	Returns a Series/DataFrame of booleans indicating if each element is in `values`.
`df.drop(labels, axis)`	Returns a DataFrame without the rows or columns named `labels` along `axis` (either 0 or 1)
`df.rename(index=None, columns=None)`	Returns a DataFrame with renamed columns from a dictionary `index` and/or `columns`
`df.sort_values(by, ascending=True)`	Returns a DataFrame where rows are sorted by the values in columns `by`
`s.sort_values(ascending=True)`	Returns a sorted Series.
`s.unique()`	Returns a NumPy array of the unique values
`s.value_counts()`	Returns the number of times each unique value appears in a Series
`pd.merge(left, right, how='inner', on='a')`	Returns a DataFrame joining DataFrames `left` and `right` on the column labeled `a`; the join is of type `inner`
`left.merge(right, left_on=col1, right_on=col2)`	Returns a DataFrame joining DataFrames `left` and `right` on columns labeled `col1` and `col2`.
`df.pivot_table(index, columns, values=None, aggfunc='mean', fill_value=None)`	Returns a DataFrame pivot table where columns are unique values from `columns` (column name or list), and rows are unique values from `index` (column name or list); cells are collected `values` using `aggfunc`. If `values` is not provided, cells are collected for each remaining column with multi-level column indexing. If a `fill_value` is provided, any `NaN` values will be replaced with that `fill_value`.
`df.set_index(col)`	Returns a DataFrame that uses the values in the column labeled `col` as the row index.
`df.reset_index()`	Returns a DataFrame that has row index 0, 1, etc., and adds the current index as a column.

Let `grouped = df.groupby(by)` where `by` can be a column label or a list of labels.

Function	Description
`grouped.count()`	Return a DataFrame containing the size of each group, excluding missing values
`grouped.size()`	Return a Series containing size of each group, including missing values
`grouped.mean()/grouped.min()/grouped.max()`	Return a Series/DataFrame containing mean/min/max of each group for each column, excluding missing values
`grouped.filter(f)` `grouped.agg(f)`	Filters or aggregates using the given function f

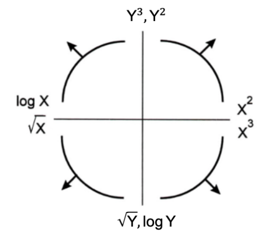
Function	Description
`s.str.len()`	Returns a Series containing length of each string
`s.str[a:b]`	Returns a Series where each element is a slice of the corresponding string indexed from `a` (inclusive, optional) to `b` (non-inclusive, optional)
`s.str.lower()/s.str.upper()`	Returns a Series of lowercase/uppercase versions of each string
`s.str.replace(pat, repl)`	Returns a Series that replaces occurences of substrings matching the regex `pat` with string `repl`
`s.str.contains(pat)`	Returns a boolean Series indicating if a substring matching the regex `pat` is contained in each string
`s.str.extract(pat)`	Returns a Series of the first subsequence of each string that matches the regex `pat`. If `pat` contains one group, then only the substring matching the group is extracted

Visualization

Matplotlib: `x` and `y` are sequences of values. `import matplotlib.pyplot as plt`

Function	Description
`plt.plot(x, y)`	Creates a line plot of `x` against `y`
`plt.scatter(x, y)`	Creates a scatter plot of `x` against `y`
`plt.hist(x, bins=None)`	Creates a histogram of `x`; `bins` can be an integer or a sequence
`plt.bar(x, height)`	Creates a bar plot of categories `x` and corresponding heights `height`

Tukey-Mosteller Bulge Diagram.



Seaborn: `x` and `y` are column names in a DataFrame `data`. `import seaborn as sns`

Function	Description
`sns.countplot(data, x)`	Create a barplot of value counts of variable `x` from `data`
`sns.histplot(data, x, stat='count', kde=False)` `sns.displot(data, x, kind='hist', rug=False, kde=False)`	Creates a histogram of `x` from `data`, where bin statistics `stat` is one of `'count'`, `'frequency'`, `'probability'`, `'percent'`, and `'density'`; optionally overlay a kernel density estimator. `displot` is similar but can optionally overlay a rug plot and/or a KDE plot
`sns.boxplot(data, x=None, y)` `sns.violinplot(data, x=None, y)`	Create a boxplot of `y`, optionally factoring by categorical `x`, from `data`. `violinplot` is similar but also draws a kernel density estimator of `y`
`sns.rugplot(data, x)`	Adds a rug plot on the x-axis of variable `x` from `data`
`sns.scatterplot(data, x, y)`	Create a scatterplot of `x` versus `y` from `data`
`sns.lmplot(x, y, data, fit_reg=True)`	Create a scatterplot of `x` versus `y` from `data`, and by default overlay a least-squares regression line
`sns.jointplot(x, y, data, kind)`	Combine a bivariate scatterplot of `x` versus `y` from `data`, with univariate density plots of each variable overlaid on the axes; `kind` determines the visualization type for the distribution plot, can be `scatter`, `kde` or `hist`

Regular Expressions

Operator	Description	Operator	Description
`.`	Matches any character except `\n`	`*`	Matches preceding character/group zero or more times
`\`	Escapes metacharacters	`?`	Matches preceding character/group zero or one times
`|`	Matches expression on either side of expression; has lowest priority of any operator	`+`	Matches preceding character/group one or more times
`\d`, `\w`, `\s`	Predefined character group of digits (0-9), alphanumerics (a-z, A-Z, 0-9, and underscore), or whitespace, respectively	`^`, `$`	Matches the beginning and end of the line, respectively
`\D`, `\W`, `\S`	Inverse sets of `\d`, `\w`, `\s`, respectively	`( )`	Capturing group used to create a sub-expression
`{m}`	Matches preceding character/group exactly `m` times	`[ ]`	Character class used to match any of the specified characters or range (e.g. `[abcde]` is equivalent to `[a-e]`)
`{m, n}`	Matches preceding character/group at least `m` times and at most `n` times. If either `m` or `n` are omitted, set lower/upper bounds to 0 and ∞, respectively	`[^ ]`	Invert character class; e.g. `[^a-c]` matches all characters except `a`, `b`, `c`

Modified lecture example for capture groups:

```
import re
lines = '169.237.46.168 - - [26/Jan/2014:10:47:58 -0800] "GET ... HTTP/1.1"'
re.findall(r'\[\d+\/(\w+)\/\d+:\d+:\d+:\d+ .+\]', line) # returns ['Jan']
```

Function	Description
`re.match(pattern, string)`	Returns a match if zero or more characters at beginning of `string` matches `pattern`, else None
`re.search(pattern, string)`	Returns a match if zero or more characters anywhere in `string` matches `pattern`, else None
`re.findall(pattern, string)`	Returns a list of all non-overlapping matches of `pattern` in `string` (if none, returns empty list)
`re.sub(pattern, repl, string)`	Returns `string` that replaces all occurrences of `pattern` with `repl`

Modeling

Concept	Formula	Concept	Formula
Variance, σ_x^2	$\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2$	Correlation r	$r=\frac{1}{n}\sum_{i=1}^{n}\frac{x_i-\bar{x}}{\sigma_x}\frac{y_i-\bar{y}}{\sigma_y}$
L_1 loss	$L_1(y,\hat{y})=\mid y-\hat{y}\mid$	Linear regression estimate of y	$\hat{y}=\theta_0+\theta_1 x$
L_2 loss	$L_2(y,\hat{y})=(y-\hat{y})^2$	Least squares linear regression	$\hat{\theta}_0=\bar{y}-\hat{\theta}_1\bar{x} \qquad \hat{\theta}_1=r\frac{\sigma_y}{\sigma_x}$
Empirical risk with loss L	$R(\theta)=\frac{1}{n}\sum_{i=1}^{n}L(y_i,\hat{y}_i)$		

Ordinary Least Squares

Multiple Linear Regression Model: $\hat{\mathbb{Y}}=\mathbb{X}\theta$ with design matrix $\mathbb{X}$, response vector $\mathbb{Y}$, and predicted vector $\hat{\mathbb{Y}}$. If there are p features plus a bias/intercept, then the vector of parameters $\theta=[\theta_0,\theta_1,\ldots,\theta_p]^T\in\mathbb{R}^{p+1}$. The vector of estimates $\hat{\theta}$ is obtained from fitting the model to the sample $(\mathbb{X},\mathbb{Y})$.

Concept	Formula	Concept	Formula
Mean squared error	$R(\theta)=\frac{1}{n}\lVert\mathbb{Y}-\mathbb{X}\theta\rVert_2^2$	Normal equation	$\mathbb{X}^T\mathbb{X}\hat{\theta}=\mathbb{X}^T\mathbb{Y}$
		Least squares estimate, if $\mathbb{X}$ is full rank	$\hat{\theta}=(\mathbb{X}^T\mathbb{X})^{-1}\mathbb{X}^T\mathbb{Y}$
Residual vector, e	$e=\mathbb{Y}-\hat{\mathbb{Y}}$	Multiple R^2 (coefficient of determination)	$R^2=\frac{\text{variance of fitted values}}{\text{variance of }y}$