Given that

$$\mathbb{E}[Z_i] = \frac{1}{6}, \mathrm{Var}[Z_i] = \frac{7}{180}$$

.

Determine $\mathbb{E}[R]$ and $\mathrm{Var}[R]$ using the properties of expectation and variance. You may give your answer in terms of the dimension d.

Basic rule of expectation and variance:

- Linearity of expectation: $\mathbb{E}[Z_i + Z_j] = \mathbb{E}[Z_i] + \mathbb{E}[Z_j]$.
- If Z_i and Z_j are independent, then $\mathrm{Var}[Z_i + Z_j] = \mathrm{Var}[Z_i] + \mathrm{Var}[Z_j]$.

(c) **[1pts]** In probability theory, one can derive that $\mathbb{P}(|Z - \mathbb{E}[Z]| \geq a) \leq \frac{\mathrm{Var}[Z]}{a^2}$ for any random variable Z. (This fact is known as Markov's Inequality.) Based on your answer to part (b), explain why does this support the claim that in high dimensions, "most points are approximately the same distance"? Let's justify this step-by-step:

i. We want to bound the probability that any given distance R is *at least* d away from its expectation. How would you mathematically write down this event, E?

ii. Use Markov's Inequality to bound $\mathbb{P}(E)$.

iii. Let d be a quantity proportional to distance (and therefore dimension). Apply the result in part (b) and note what happens to $\mathbb{P}(E)$ as d goes to ∞.

2. **[8pts] Decision Trees.** *This question is taken from a project by Lisa Zhang and Michael Guerzhoy.*

In this question, you will use the `scikit-learn` decision tree classifier to classify real vs. fake news headlines. The aim of this question is for you to read the `scikit-learn` API and get comfortable with training/validation splits.

We will use a dataset of 1298 "fake news" headlines (which mostly include headlines of articles classified as biased, etc.) and 1968 "real" news headlines, where the "fake news" headlines are from https://www.kaggle.com/mrisdal/fake-news/data and "real news" headlines are from https://www.kaggle.com/therohk/million-headlines. The data were cleaned by removing words from fake news titles that are not a part of the headline, removing special characters from the headlines, and restricting real news headlines to those after October 2016 containing the word "trump". For your interest, the cleaning script is available as `clean_script.py` on the course web page, but you do not need to run it. The cleaned-up data are available as `clean_real.txt` and `clean_fake.txt` on the course web page.

Each headline appears as a single line in the data file. Words in the headline are separated by spaces, so just use `str.split()` in Python to split the headlines into words.

You will build a decision tree to classify real vs. fake news headlines. Instead of coding the decision trees yourself, you will do what we normally do in practice — use an existing implementation. You should use the `DecisionTreeClassifier` included in `sklearn`. Note that figuring out how to use this implementation is a part of the assignment.

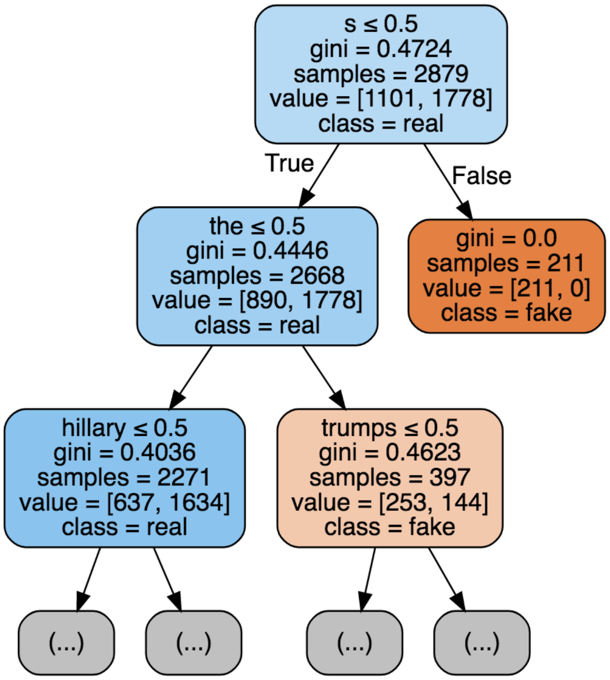
Here's a link to the documentation of `sklearn.tree.DecisionTreeClassifier`: http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

All code should be included in the file `hw1_code.py` which you submit through MarkUs.

(a) **[2pt]** Write a function `load_data` which loads the data, preprocesses it using a vectorizer (http://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text), and splits the entire dataset randomly into 70% training, 15% validation, and 15% test examples.

(b) **[2pt]** Write a function `select_model` which trains the decision tree classifier using at least 5 different values of `max_depth`, as well as three different split criteria (information gain, log loss and Gini coefficient), evaluates the performance of each one on the validation set, and prints the resulting accuracies of each model. You should use `DecisionTreeClassifier`, but you should write the validation code yourself. In your solution PDF (`hw1_writeup.pdf`), include the output of this function as well as a plot of the validation accuracy vs. `max_depth`.

(c) **[1pt]** Now let's stick with the hyperparameters which achieved the highest validation accuracy. Extract and visualize the first two layers of the tree. Your visualization may look something like what is shown below, but it does not have to be an image: it is perfectly fine to display text. It may be hand-drawn. Include your visualization in your solution PDF (`hw1_writeup.pdf`).



(d) **[3pts]** Write a function `compute_information_gain` which computes the information gain of a split on the training data. That is, compute $I(Y, x_i)$, where Y is the random variable signifying whether the headline is real or fake, and x_i is the keyword chosen for the split.

Report the outputs of this function for the topmost split from the previous part, and for several other keywords.

3. **[8pts] Regularized Linear Regression.** For this problem, we will use the linear regression