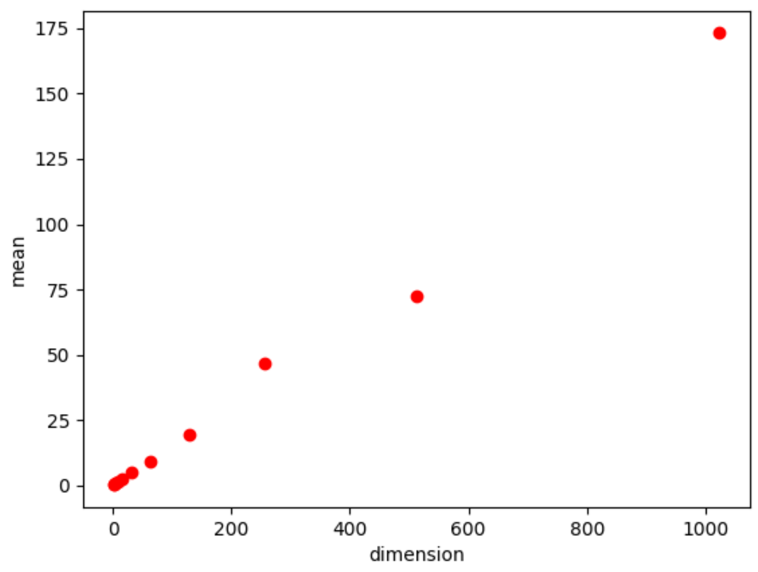
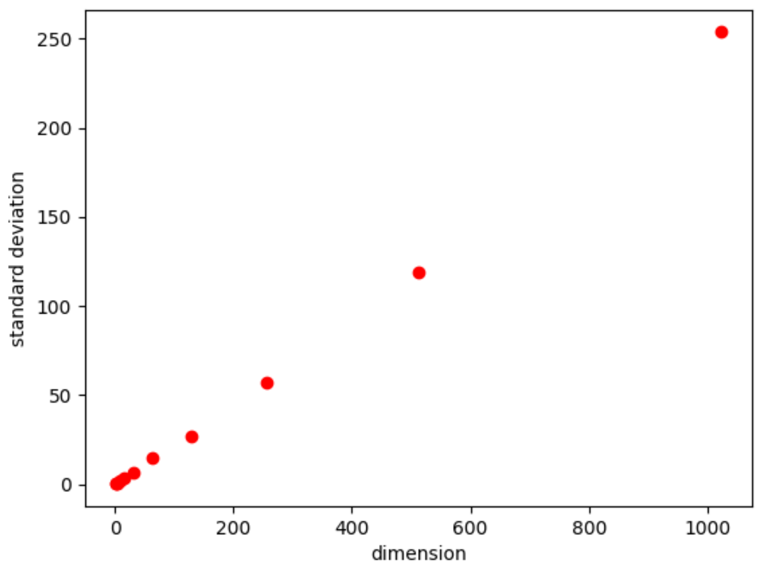
Q1

a) graphs for mean and standard deviation:





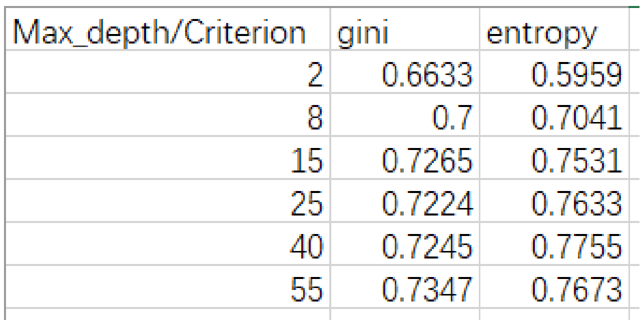
b) $E[R] = \dfrac{d}{6}$ and $\text{Var}[R] = \dfrac{7d}{180}$

c) Markov's Inequality suggests that probability of euclidean distance deviating from expected value very far is unlikely. So that the euclidean distance between two points is likely to be similar (approximately the same distance), and on high dimensions, value of d is large, so $E[R]$ is large. So most points are far away in high dimensions.

Q2

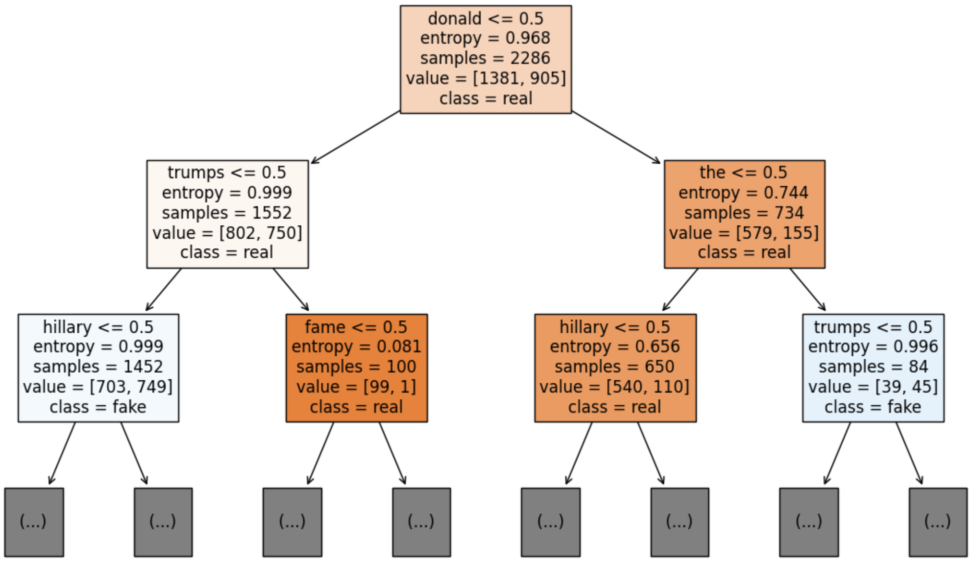
a) see python file

b) results as followed



Max_depth/Criterion	gini	entropy
2	0.6633	0.5959
8	0.7	0.7041
15	0.7265	0.7531
25	0.7224	0.7633
40	0.7245	0.7755
55	0.7347	0.7673

c) tree as followed



d) At topmost split, the information gain is 0.0513, with keyword "trump" and class equals real, the information gain is 0.0591.

With the mutual_inf_classif function in sklearn, there is a clear trend of information gain decreasing as we traverse the keyword from the decision tree downwards.

Q3

a) Apply definition of gradient descent:

$$
\begin{aligned}
w_j \leftarrow w_j - \alpha \frac{\partial J_{reg}^{\beta}}{\partial \omega_j} \\
&= \omega_j - \alpha \cdot (\frac{\partial \mathcal{J}}{\partial \omega_j} + \frac{\partial \mathcal{R}}{\partial \omega_j}) \\
&= \omega_j - \alpha[\frac{1}{2N}\sum_{i=1}^{N} \frac{\partial}{\partial \omega_j}(y^{(i)} - t^{(i)})^2 + \beta_j \omega_j] \\
&= \omega_j - \alpha[\frac{1}{N}\sum_{i=1}^{N} x_j(y^{(i)} - t^{(i)}) + \beta_j \omega_j] \\
&= (1 - \alpha\beta_j)\omega_j - \frac{\alpha}{N}\sum_{i=1}^{N} x_j(y^{(i)} - t^{(i)}) \\
b \leftarrow b - \alpha \frac{\partial J_{reg}^{\beta}}{\partial b} \\
&= b - \alpha \cdot (\frac{\partial \mathcal{J}}{\partial b} + \frac{\partial \mathcal{R}}{\partial b}) \\
&= b - \alpha[\frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - x^{(i)}) + 0] \\
&= b - \frac{\alpha}{N}\sum_{i=1}^{N}(y^{(i)} - x^{(i)})
\end{aligned}
$$

In this method, we adjust the weight in the direction of steepest descent, we multiply α by the weights to prevent it from changing to fast. Therefore the weight is "decaying" to decrease the cost function to minimize the cost function.

b) Use formula we get from part a)

$$\begin{aligned}
\frac{\partial J_{reg}^{\beta}}{\partial \omega_j} &= \frac{1}{N}\sum_{i=1}^{N} x_j^{(i)}(y^{(i)} - t^{(i)}) + \beta_j \omega_j \\
&= \frac{1}{N}\sum_{i=1}^{N} x_j^{(i)}(\sum_{j'=1}^{D} \omega_{j'} x_{j'}^{(i)}) - \frac{1}{N}\sum_{i=1}^{N} x_j^{(i)} t^{(i)} + \beta_j \omega_j && \text{(expand } y\text{)} \\
&= \sum_{j'=1}^{D} \omega_{j'}(\frac{1}{N}\sum_{i=1}^{N} x_j^{(i)} x_{j'}^{(i)} + \beta_j(\text{if } \omega_j = \omega_{j'})) - \frac{1}{N}\sum_{i=1}^{N} x_j^{(i)} t^{(i)} && \text{(take out } \omega_{j'}\text{)} \\
&= \sum_{j'=1}^{D} \omega_{j'}(\frac{1}{N}\sum_{i=1}^{N} x_j^{(i)} x_{j'}^{(i)} + \beta_j \Gamma(\omega_j)) - \frac{1}{N}\sum_{i=1}^{N} x_j^{(i)} t^{(i)}
\end{aligned}$$

with

$$\Gamma(\omega_j) = \begin{cases} 1 & j' = j \\ 0 & \text{otherwise} \end{cases}$$

We get $A_{jj'} = \frac{1}{N}\sum_{i=1}^{N} x_j^{(i)} x_{j'}^{(i)} + \beta_j \Gamma(\omega_j)$ and $c_j = \frac{1}{N}\sum_{i=1}^{N} x_j^{(i)} t^{(i)}$

c) $\mathbf{A} = \frac{1}{N}\mathbf{X}^\top \mathbf{X} + \lambda D\beta$ where D is the diagonal matrix
$\mathbf{c} = \frac{1}{N}\mathbf{X}^\top \mathbf{t}$
Note we have $\mathbf{A}\mathbf{w} = \mathbf{c}$ from b), expand the equation, we get

$$(\frac{1}{N}\mathbf{X}^\top \mathbf{X} + \lambda D\beta)\mathbf{w} = \frac{1}{N}\mathbf{X}^\top \mathbf{t}$$

Rearrange to get:

$$\mathbf{w} = \mathbf{X}^\top \mathbf{t}(\mathbf{X}^\top \mathbf{X} + \lambda N D\beta)^{-1}$$