Problem 1

(a) The Hoeffding inequality can be directly used because $P(Z_i = 1) = P\left(|\hat{\phi}_i - \phi_i| > \gamma\right)$ where $\hat{\phi}_i$ is the mean of m Bernoulli distributed random variables. Let $\psi_i = P(Z_i = 1)$, then

$$\begin{aligned}\psi_i &= P(Z_i = 1) \\ &= P\left(|(\hat{\phi}_i - \phi_i| > \gamma\right) \\ &\leq 2e^{-2\gamma^2 m}\end{aligned}$$

(b) Let us prove the statement by induction. For the first step of the induction let us prove that it is true for some k, and we choose $k = 1$. Then

$$\begin{aligned}\text{for } t < 0, \quad & P(V_1 > t) = 1 = P(W_1 > t) \\ \text{for } t \geq 1, \quad & P(V_1 > t) = 0 = P(W_1 > t) \\ \text{for } 0 \leq t < 1, \quad & P(V_1 > t) = P(V_1 = 1) = E[V_1] < E[W_1] = P(W_1 = 1) = P(W_1 > t)\end{aligned}$$

Where the first equality occurs because V_1 and W_1 can only take values 0 and 1, and therefore are always greater than a negative number. The second equality follows because the maximum values of V_1 and W_1 is 1, so it can never be greater than 1. In the third statement, $P(V_1 > t) = P(V_1 = 1)$ because V_1 is binary valued and hence greater than t only when it takes the value $V_1 = 1$. The inequality comes from the assumption we are given. **Hence**, the statement is true for some $k = 1$.
Now let us assume that it is true for some $k = n - 1$, i.e.

$$P(\sum_{i=1}^{n-1} V_i > t) \leq P(\sum_{i=1}^{n-1} W_i > t) \tag{1}$$

We must show that it is then true for $k = n$. So

$$\begin{aligned}P\left(\sum_{i=1}^{n} V_i > t\right) &= P\left(\sum_{i=1}^{n-1} V_i + V_n > t\right) \\ &= P\left(\sum_{i=1}^{n-1} V_i > t\right) P\left(V_n = 0\right) + P\left(\sum_{i=1}^{n-1} V_i > t - 1\right) P\left(V_n = 1\right) \\ &= P\left(\sum_{i=1}^{n-1} V_i > t\right) \left[1 - P(V_n = 1)\right] + P\left(\sum_{i=1}^{n-1} V_i > t - 1\right) P\left(V_n = 1\right) \\ &= P\left(\sum_{i=1}^{n-1} V_i > t\right) + \left[P\left(\sum_{i=1}^{n-1} V_i > t - 1\right) - P\left(\sum_{i=1}^{n-1} V_i > t\right)\right] P\left(V_n = 1\right) \\ &\leq P\left(\sum_{i=1}^{n-1} V_i > t\right) + \left[P\left(\sum_{i=1}^{n-1} V_i > t - 1\right) - P\left(\sum_{i=1}^{n-1} V_i > t\right)\right] P\left(W_n = 1\right)\end{aligned}$$

Where in going from the first to the second line we have considered the two possible values of V_n to re-write the probability, while in going from the second to last to the last line, we used the fact that $P(V_n = 1) \leq P(W_n = 1)$ along with the fact that $\left[P\left(\sum_{i=1}^{n-1} V_i > t - 1\right) - P\left(\sum_{i=1}^{n-1} V_i > t\right)\right]$ is necessarily positive. We

Problem 1 continued on next page...

now continue:

$$\begin{aligned}
P\left(\sum_{i=1}^{n} V_i > t\right) &\leq P\left(\sum_{i=1}^{n-1} V_i > t\right) + \left[P\left(\sum_{i=1}^{n-1} V_i > t-1\right) - P\left(\sum_{i=1}^{n-1} V_i > t\right)\right] P\left(W_n = 1\right) \\
&= P\left(\sum_{i=1}^{n-1} V_i > t\right) P(W_n = 0) + P\left(\sum_{i=1}^{n-1} V_i > t-1\right) P\left(W_n = 1\right) \\
&\leq P\left(\sum_{i=1}^{n-1} W_i > t\right) P(W_n = 0) + P\left(\sum_{i=1}^{n-1} V_i > t-1\right) P\left(W_n = 1\right) \\
&\leq P\left(\sum_{i=1}^{n-1} W_i > t\right) P(W_n = 0) + P\left(\sum_{i=1}^{n-1} W_i > t-1\right) P\left(W_n = 1\right) \\
&= P\left(\sum_{i=1}^{n-1} W_i + W_n > t\right) \\
&= P\left(\sum_{i=1}^{n} W_i > t\right)
\end{aligned}$$

where in lines 2 to 3 and 3 to 4 we have used the inductive Hypothesis (Eq. 1). Thus we have showed that the statement is true for some $k = 1$ and that truth for $k = n - 1$ implies the truth for $k = n$. Thus the statement is proved by induction.

(c) The simplest way to provide a bound is to use Markov's inequality which states: let X be a non-negative random variable, and a a positive constant, then

$$P(X \geq a) \leq \frac{E[X]}{a}$$

Therefore,

$$\begin{aligned}
P\left(\frac{1}{n}\sum_{i=1}^{n} Z_i \geq \tau\right) &\leq \frac{E\left[\frac{1}{n}\sum_{i=1}^{n} Z_i\right]}{\tau} \\
&= \frac{\frac{1}{n}\sum_{i=1}^{n} E\left[Z_i\right]}{\tau} \\
&\leq \frac{\frac{1}{n}\sum_{i=1}^{n} 2e^{-2\gamma^2 m}}{\tau} \\
&= \frac{2}{\tau} e^{-2\gamma^2 m}
\end{aligned}$$

where from the first to the second line we use the fact that the expected value of the linear sum is a sum of the expected values, while from second to the third line we have used part (a) of this problem (Hoeffding inequality).

Problem 2

The answer is $VC(\mathcal{H}) = d + 1$.
Proof: The key insight is that a polynomial of with d roots (i.e. d crossings of the line $h_\theta(x) = 0$) can partition the real number line into $d+1$ segments, and hence can classify $d+1$ points. Let us try to be more specific:

First let us show that $\mathcal{H} = \{h_\theta : \theta \in \mathbb{R}^{d+1}\}$ can shatter a set of $d + 1$ points where the inputs $\mathcal{X} = \mathbb{R}$. For example, place the $d + 1$ points equally spaced on the real number line on every even number (i.e. there are $d + 1$ points, at positions $x = 0, 2, 4, 6 \ldots, 2d$. Consider the most extreme labelling of these $d + 1$ points where the labelling varies at $+1, -1, +1, -1 \ldots$. Now choose the polynomial (i.e. choose $\theta's$) such that the d roots of this polynomial at positions $x_i = 1, 3, 5, \ldots, 2d + 1$, i.e. the polynomial takes the form $h_\theta(x) = \pm\Pi_{i=1}^{d}(x - (2i + 1))$, where the overall sign depends on the classification of the first point. It is clear that this form of $h_\theta(x)$ correctly classifies the set for the labelling I have described above.

Now for any other labelling, e.g. when two consecutive points have the same label, we simply remove one of the roots described above. We always place roots of the polynomial between distinct groups of points - this is always possible provided there are only $d + 1$ points. Hence for any labelling of the set of $d + 1$ points, we can correctly classify the set, i.e. $\mathcal{H}$ shatters the set of $d + 1$ points.

To complete the proof we must show why $\mathcal{H}$ cannot shatter a set of $d + 2$ points on the real number line. To see why, consider the above set of $d + 1$ points at positions $(0, 2, 4, 6 \ldots)$ with labelling $(+1, -1, +1, -1, \ldots)$. Now add one more point at position $x = -2$, with labelling -1. It is clear that a polynomial of degree d can no longer classify this set, because there is no root we can add, i.e. ($h_\theta(x)$ cannot cross the real number line more than d times. Thus, $VC(\mathcal{H}) < d + 2$.

Problem 3

(a) The bound follows from the fact that the decision boundary for an SVM only changes when we remove a support vector from our training set (this is only true when the data is linearly separable). Thus, for a given partitioning of the set $S = S_{\text{train}} \cup S_{\text{test}}$ where S_{test} contains 1 point, and S_{train} contains $m-1$ points, we only make an error when S_{test} contains a support vector. Now there are m possible partitions of the data, and during these m trials, the most number of errors we can make corresponds to the number of support vectors $|SV|$. Thus,

$$\hat{\varepsilon}_{\text{LOOCV}} \leq \frac{|SV|}{m}$$

To provide a more formal proof, we can examine what happens to the true decision boundary $w^T x + b = 0$ (obtained from training on the full set S) upon leaving out one point. Let us consider how to relate the true w's, b's and α's in the primal and dual optimization problem to those obtained when we train on the set S_{train}, i.e. $\hat{w}$'s, $\hat{b}$'s and $\hat{\alpha}_i$'s. If the point that we have omitted is not a support vector, we obtain the we have $\hat{\alpha}_i = \alpha_i$ for all i. This means that we obtain the same $\hat{w} = w$ and $\hat{b} = b$. Thus the same decision boundary means that we will classify the extra point correctly.
Leaving out a support vector from our training set leads to differing $\alpha's$, which in turn means that $\hat{w} \neq w$ and $\hat{b} \neq b$. We will therefore incorrectly classify the test point. For m different paritions, there are at most $|SV|$ times that we can incorrectly obtain α's , w's and b's, so the error will be bounded by $|SV|/m$.

(b) So long as the data remains linearly separable in the higher dimensional feature space (as we are told in the problem), the same arguments written above apply. The essential point is that because the data is linearly separable, we can only make an error when we omit a support vector in the higher dimensional feature space. The same arguments as above follow through.

Problem 4

First notice that the prior for θ can be written as

$$p(\theta) = \frac{1}{(2\pi)^{\frac{n+1}{2}}\tau} e^{-||\theta||_2^2}$$

Now from the definition of θ_{MAP} we necessarily have

$$p(\theta_{\text{MAP}}) \prod_{i=1}^{m} p\left(y^{(i)}|x^{(i)};\theta_{\text{MAP}}\right) \geq p(\theta_{\text{ML}}) \prod_{i=1}^{m} p\left(y^{(i)}|x^{(i)};\theta_{\text{ML}}\right)$$

because θ_{MAP} maximizes the function $p(\theta)\prod_{i=1}^{m} p(y^{(i)}|x^{(i)};\theta)$. So we have

$$\frac{p(\theta_{\text{MAP}})}{p(\theta_{\text{ML}})} \geq \frac{\prod_{i=1}^{m} p\left(y^{(i)}|x^{(i)};\theta_{\text{ML}}\right)}{\prod_{i=1}^{m} p\left(y^{(i)}|x^{(i)};\theta_{\text{MAP}}\right)}$$

but because θ_{ML} maximizes $\prod_{i=1}^{m} p\left(y^{(i)}|x^{(i)};\theta\right)$, the right hand side is greater than or equal to 1. Thus, we have

$$\begin{aligned}
\frac{p(\theta_{\text{MAP}})}{p(\theta_{\text{ML}})} &\geq 1 \\
\implies \exp\frac{1}{2\tau^2}\left[||\theta_{\text{ML}}||_2^2 - ||\theta_{\text{MAP}}||_2^2\right] &\geq 1 \\
\implies ||\theta_{\text{ML}}||_2^2 - ||\theta_{\text{MAP}}||_2^2 &\geq 0 \\
\text{i.e.} \qquad ||\theta_{\text{ML}}||_2 &\geq ||\theta_{\text{MAP}}||_2,
\end{aligned}$$

as required.

Problem 5

(a) The KL divergence is

$$\begin{aligned} KL(P||Q) &= \sum_x P(x) \log \frac{P(x)}{Q(x)} \\ &= \sum_x P(x) \left\{ -\log \left(\frac{Q(x)}{P(x)} \right) \right\} \\ &= E\left[-\log \left(\frac{Q(x)}{P(x)} \right) \right] \\ &\geq -\log \left\{ E\left[\frac{Q(x)}{P(x)} \right] \right\} \\ &= -\log \left\{ \sum_x P(x) \frac{Q(x)}{P(x)} \right\} \\ &= -\log \left\{ \sum_x Q(x) \right\} \\ &= -\log(1) \\ &= 0 \\ \implies \quad KL(P||Q) &\geq 0 \end{aligned}$$

Because $-\log x$ is strictly convex, $KL(P||Q) = 0$, i.e. the equality above occurs **only when**

$$\begin{aligned} \frac{Q(x)}{P(x)} &= E\left[\frac{Q(x)}{P(x)} \right] \\ &= \sum_x P(x) \frac{Q(x)}{P(x)} \\ &= \sum_x Q(x) \\ &= 1 \\ \implies \quad Q(x) &= P(x) \end{aligned}$$

(b) We begin with the definition of the KL divergence, and recall that most fundamental of probability identities: $P(x,y) = P(x)P(y|x) = P(y)P(x|y)$.

$$\begin{aligned} KL\left(P(X,Y)||Q(X,Y)\right) &= \sum_x \sum_y P(x,y) \log \frac{P(x,y)}{Q(x,y)} \\ &= \sum_{x,y} P(x)P(y|x) \log \frac{P(x)P(y|x)}{Q(x)Q(y|x)} \\ &= \sum_{x,y} P(x)P(y|x) \log \frac{P(x)}{Q(x)} + \sum_{x,y} P(x)P(y|x) \log \frac{P(y|x)}{Q(y|x)} \end{aligned}$$

now note that $\sum_y P(x)P(y|x) \equiv \sum_y P(x,y) = P(x)$. So we find

$$\begin{aligned} KL\left(P(X,Y)||Q(X,Y)\right) &= \sum_x P(x) \log \frac{P(x)}{Q(x)} + \sum_x P(x) \sum_y P(y|x) \log \frac{P(y|x)}{Q(y|x)} \\ &= KL\left(P(X)||Q(X)\right) + KL\left(P(Y|X)||Q(Y|X)\right) \end{aligned}$$

(c) Finally, let us write the KL divergence between $\hat{P}$ and P_θ:

$$\begin{aligned}
\arg\min_\theta KL(\hat{P}||P_\theta) &= \arg\min_\theta \sum_x \hat{P}(x) \log \frac{\hat{P}(x)}{P_\theta(x)} \\
&= -\arg\min_\theta \sum_x \hat{P}(x) \log P_\theta(x) \\
&= \frac{1}{m} \arg\max_\theta \sum_x \sum_{i=1}^m 1\{x^{(i)} = x\} \log P_\theta(x) \\
&= \frac{1}{m} \arg\max_\theta \sum_{i=1}^m \log P_\theta(x^{(i)}) \\
&\equiv \arg\max_\theta \sum_{i=1}^m \log P_\theta(x^{(i)})
\end{aligned}$$

where we have used the indicator function to collapse the sum over x (i.e. the indicator function picks $x = x^{(i)}$ in the sum over x), and the factor of $1/m$ does not affect the arg max.

Problem 6

The raw and compressed images are shown in Figs. 1 and Fig. 2. Matlab code is attached to this document. We can compute the compression factor by noting that a single pixel has 16 possible values and so is represented by a single 4 bit integer. Previously, the image contained 3×8 bits per pixel. Thus, the compression is $\approx 24/4 = 6$ (note that we still have to store a 16×3 matrix with each entry containing 8bit numbers.

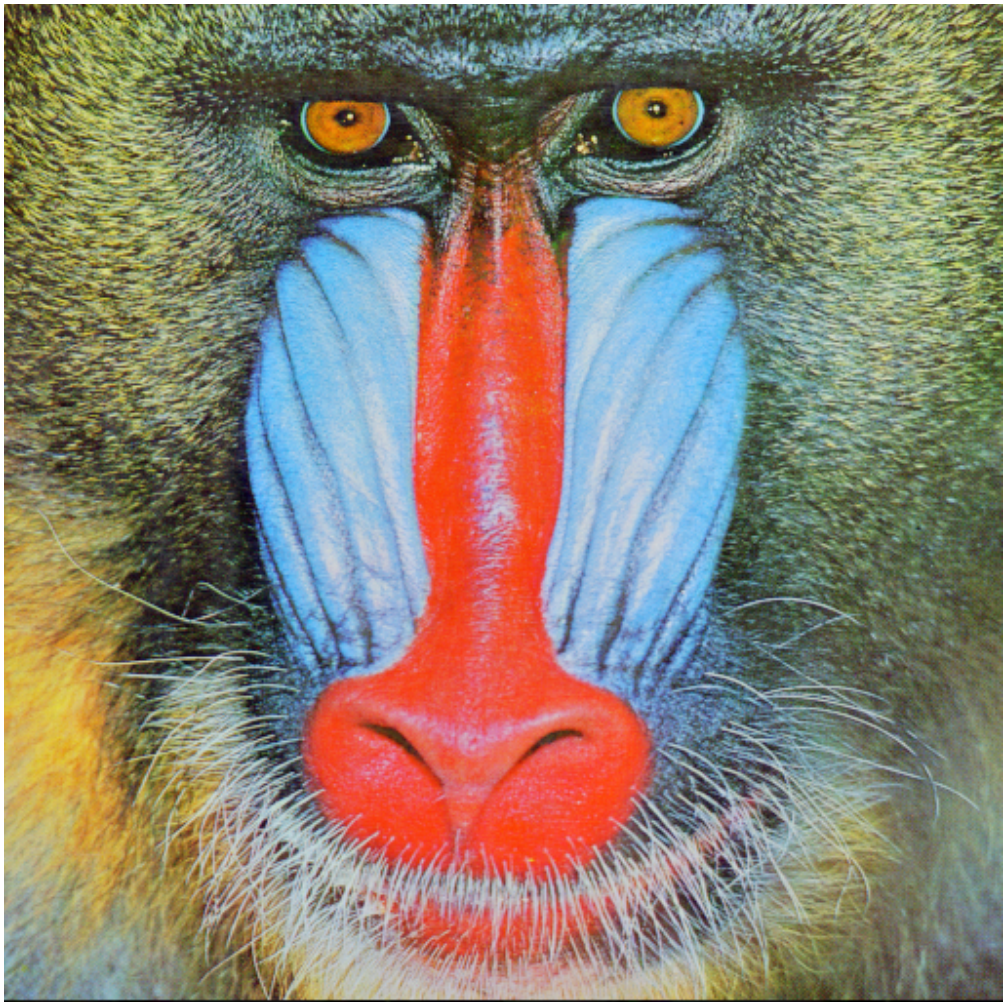

Figure 1: The raw (uncompressed) Mandrill image

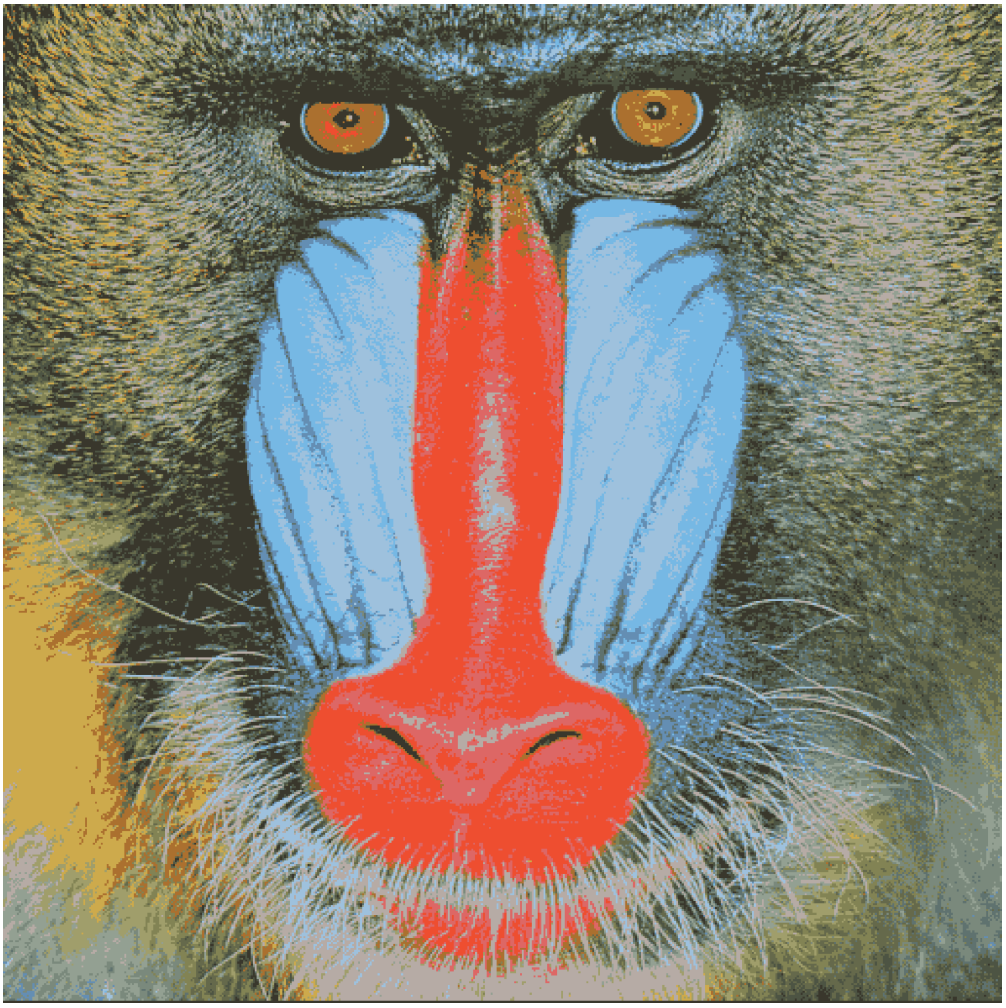

Figure 2: The compressed Mandrill image