# Assignments:
# Foundations of Machine Learning

Instructor: Saketh

December 6, 2022

**NOTE:** Never submit CODE

## 1 Set-1: Due on 21-Aug-2022 11:59pm

### 1.1 Probability Basics

1. From first principles, prove the following in three cases: when both random variables are discrete, when they are jointly continuous, and when one of them is continuous and the other is discrete.
   (a) Bayes rule for random variables: $p(y/x) = \frac{p(x/y)p(y)}{p(x)}$.
   (b) Total expectation rule: $E[f(X,Y)] = E\left[E\left[f(X,Y)/X\right]\right] = E\left[E\left[f(X,Y)/Y\right]\right]$.
2. Suppose that the expected number of accidents per week on the highway through IITH is four. Suppose also that the numbers of commuters injured in each accident are independent random variables with a common mean of 2. Assume also that the number of commuters injured in each accident is independent of the number of accidents that occur. What is the expected number of injuries during a week on that highway? Compute this formally using the total expectation rule.
3. Let $p(x,y)$ be a (multivariate) Gaussian density. Assume dimensionalty of $x, y$ is respectively $n, m$. Prove that marginals $p(x), p(y)$ will again be some Gaussian densities. Prove that the conditionals $p(y/x), p(x/y)$ will again be some Gaussian densities (for any given value of the conditioned variable).
4. Let $Y = WX + b$, where $W \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^{m\times 1}$ are fixed and known. Express the mean vector and covariance matrix of $Y$ in terms of those for $X$.

### 1.2 Bayes Optimal

1. Find the Bayes Optimal function corresponding to:
   (a) $\mathcal{X} = \mathbb{R}, \mathcal{Y} = \{0, 1\}$ and $p^*(1/x) = \frac{1}{1+e^{x^2-5x+6}}$ and 0-1 loss.
   (b) $\mathcal{X} = \mathbb{R}, \mathcal{Y} = \{0, 1\}$ and $p^*(1/x) = \frac{1}{1+e^{x^2-5x+6}}$ and loss defined by $l(0,0) = l(1,1) = 0$ and $l(0,1) = 0.5, l(1,0) = 2$.
   (c) $\mathcal{X} = \mathbb{R}^n_+, \mathcal{Y} = \mathbb{R}_{++}$ and $\Lambda(x) = maxeig(\sum_{i=1}^n A_i x_i + A_0),\ x \geq 0$, where $A_1, \ldots, A_n$ are some psd (positive semi-definite) matrices, $A_0$ is a pd (positive definite) matrix and $maxeig$ is the maximum Eigenvalue function. Consider square loss and $p^*(y/x) \propto e^{-\Lambda(x)y}, y > 0$.
   (d) $\mathcal{X} = \mathbb{R}^n_+, \mathcal{Y} = \mathbb{R}_{++}$ and $\Lambda(x) = maxeig(\sum_{i=1}^n A_i x_i + A_0),\ x \geq 0$, where $A_1, \ldots, A_n$ are some psd (positive semi-definite) matrices, $A_0$ is a pd (positive definite) matrix and $maxeig$ is the maximum Eigenvalue function. Consider $p^*(y/x) \propto e^{-\Lambda(x)y}, y > 0$ and absolute deviation loss defined by $l(y,z) \equiv |y - z|$.

## 2 Set-2: Due on 11-Sep-2022 11:59pm

### 2.1 Linear Regression

1. Exercises 9.1, 9.2 in [Shalev-Shwartz and Ben-David(2014)].

2. Consider a regression problem where the label space is $R^d$. Consider the natural extension of the linear model studied in lecture: functions $f : \mathbb{R}^n \mapsto \mathbb{R}^d$ of the form $f(x) \equiv W^\top \phi(x)$, where $W$ is a $n \times d$ matrix. Show that performing linear regression with squared-loss using this model is essentially SAME as solving $d$ classical linear regression problems over real labels.

3. Consider the regression dataset at `https://archive.ics.uci.edu/ml/datasets/DrivFace`[1]. Process this data in standard format of input-label pairs. Then split this dataset into two random halves. Use the first as training and second as the test set. Report the explained variance on test set for regressor obtained using `scikit` Linear Regression class[2] trained using the training half. In this case, note that the inputs are already in Euclidean space, so please try the identity feature map i,e, $\phi(x) = x$. Also try[3], $\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$. It is enough if you submit these two numbers for the explained variance obtained with the given feature maps. Please do NOT submit any code.

[1] Ignore the classification and clustering task's data

[2] At `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html`

[3] Here, $x^2$ denotes the vector whose entries are square of those in vector $x$.

4. This is a simulation based problem:

   (a) Generate $2m$ samples from your favourite $d$-dimensional distribution (say d-dim multivariate Gaussian).

   (b) With each such 'input' sample, $x$, obtain a 'label', $y$, by sampling from $\mathcal{N}({w^*}^\top x, 1)$. Here, $w^*$ is your favourite (any) vector! This will create $2m$ pairs of input-labels. Arbitrarily call $m$ of these as the training set $\mathcal{D}$ and the remaining as the test set $\mathcal{T}$.

   (c) Consider the Linear Regression set-up. Convince yourself that $w^*$ is the Bayes optimal's parameter.

   (d) Solve the ERM (least squares) problem using EITHER of the following: (a) Solve the normal equation using `numpy.linalg.lstsq`[4] or (b) Solve using `Linear Regression` class in `scikit`[5]. For a fixed $m$, submit plot of explained variance[6] on your test set vs $d$. If your solver fails to converge in some cases, then report such cases. It is enough if you submit this one plot and the list of convergence failed cases (if any). Please do NOT submit any code.

## 2.2 Classification with Linear Models

1. Consider the binary classification dataset at `https://archive.ics.uci.edu/ml/datasets/LSVT+Voice+Rehabilitation`. Process this data in standard format of input-label pairs. Then split this dataset into two random halves. Use the first as training and second as the test set. Report the classifcation accuracy on test set for the classifier obtained using (both of the following): (a) `scikit` Perceptron class[7] (b) Logistic Regression class[8], trained using the training half. In this case, note that the inputs are already in Euclidean space, so please try the identity feature map i,e, $\phi(x) = x$. Also try[9], $\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$. It is enough if you submit these four numbers for the classification accuracies obtained with the given feature maps and trained classifiers. Please do NOT submit any code.

[4] At `https://numpy.org/doc/stable/reference/generated/numpy.linalg.lstsq.html`.

[5] At `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html`

[6] At `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.explained_variance_score.html`.

[7] At `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html`

[8] At `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`. PLEASE SET option 'penalty' to 'none'.

[9] Here, $x^2$ denotes the vector whose entries are square of those in vector $x$.

### 2.3 Gradient Descent & SGD

1. In this problem, your goal is solve $\min_{v \in \mathbb{R}^{10}} f(v)$, where $f(v) = v^\top A v - 2b^\top v + c$. Generate $A$ randomly using `sklearn.datasets.make_spd_matrix`, and generate $b$ and $c$ using `np.random.rand` function of `numpy`. Once you obtain $A, b, c$, keep them fixed for the entire problem. Solve this min. problem using the following different methods:

   (a) Analytically, by setting gradient to zero. (You can verify that $f$ is convex).

   (b) Numerically using Gradient descent: Initialize $v = [\frac{1}{10}, \frac{1}{10}, \ldots, \frac{1}{10}]$. Step-size for Gradient Descent: $l_{GD} = \frac{1}{2\|A\|+\|b\|_2}$ where $\|A\|$ is the spectral norm of A. The Spectral norm of matrix $A$ can be computed using `np.linalg.norm(A, ord=2)`. The norm of $b$ can be computed using `np.linalg.norm(b)`. 1000 iterations.

   (c) Numerically using SGD: Initialize $v = [\frac{1}{10}, \frac{1}{10}, \ldots, \frac{1}{10}]$. At every iteration, instead for using the computed gradient directly, perturb it with noise. This will simulate a random direction. More specifically, at every iteration, $0.5 * \epsilon$ is added to the gradient vector where $\epsilon_i \sim \mathcal{N}(0, 1)$ for $i \in [10]$. A random Gaussian noise can be generated using `np.random.randn()`. Use step size $\frac{1}{100} l_{GD}$ where $l_{GD}$ is the step size used for Gradient Descent. run for 100000 iterations.

   Make a plot of iterations (x-axis) vs objective value (at that iteration; y-axis) with gradient descent. In the same figure plot the same for SGD. And draw a horizontal line at the optimal objective value obtained using the analytical solution. Ideally, GD and SGD must converge to the horizontal line as iterations increase. Submit this plot. Please do NOT submit any another detail.

## 3 Set-3: Due on 2-Oct-2022 11:59pm

### 3.1 MLE

1. Exercises 4.5a-c in [Murphy(2022)], exercises 3.6,3.8,3.11a,3.11b, in [Murphy(2012)].
2. Exercise 24.2 (only first of the three parts) in [Shalev-Shwartz and Ben-David(2014)].

### 3.2 Generative models for Regression

1. Analogous to your finding in problem 2 in section 2.1 above, prove that the generative linear regression set-up with $d$-dimensional labels will also degenerate into $d$ generative linear scalar regression problems. Hint: Use the Schur complement lemma to rewrite the result in lecture, which was in terms of precision matrices, to one where covariance matrices are used. Then observe the final simplified expression.

2. Consider a generative model for regression problems where $p^*(y)$ is modelled using a Gaussian model i.e., $\mathcal{N}(\mu_2, \Sigma_2)$, where $\mu_2 \in \mathbb{R}^d, \Sigma_2 \succ 0$ are the parameters. And, $p^*(x/y)$ is modelled using: $\mathcal{N}(W^\top y + b, \Sigma_1)$, where $W \in \mathbb{R}^{d \times n}, b \in \mathbb{R}^n, \Sigma_1 \succ 0$ are the parameters. Show that $p^*(y/x)$ will be a Gaussian. Express this Gaussian's mean and covariance in terms for $\mu_2, W, b, \Sigma_2, \Sigma_1$. This model (for the joint likelihood $p^*(x, y)$) is called as the linear Gaussian system. Do you think this model is also plagued with the degeneration issue proved in problem 2 (section 2.1) and problem 1 (section 3.2)? In case you can't answer this question theoretically, then you may use the simulation set-up in the immediate following question to answer the same.

3. Split the multi-regression[10] dataset: `https://archive.ics.uci.edu/ml/datasets/Residential+Building+Data+Set` into two random halfs and call one as train and the other as test. Using the training half, train the model in the above question (linear Gaussian system), train the generative linear regression's model, and train the linear regression's model[11]. Predict the 2-dimensional label vectors over the test half using the 3 trained models. Report the trained model parameters along with the explained variance, for each of the two labels, obtained over the test half in all the three cases. Include the indices of the training half in your report[12].

## 3.3 Bayes Classifier

1. Exercises 3.20, 3.22, 4.18-4.23 in [Murphy(2012)].

## 3.4 Discriminative linear regression

1. Exercise 7.5,7.6,7.9 [Murphy(2012)].

## 3.5 Multiclass Logistic Regression

1. Exercise 10.1, 10.3 [Murphy(2022)].

2. Consider the 3-class classification dataset `https://archive.ics.uci.edu/ml/datasets/Parkinson+Disease+Spiral+Drawings+Using+Digitized+Graphics+Tablet#`. Produce a visualization, similar to fig 9.2 in [Murphy(2022)], of the classifier obtained with the following methods: a) Bayes classifier b) Bayes classifier with tied covariances c) Naive Bayes classifier d) Naive

[10] The label here is in $\mathbb{R}^2$.

[11] Code for linear regression is at `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html`. In view of problem 2 in section 2.1 above, training linear regression model separately for each of the two labels is the way. For the other two models write your own (simple) code and the models naturally can handle 2-dim labels.

[12] We can then easily verify the correctness of your code, without the need for actually seeing your code. Secondly, the chance that two students happen to pick the same indices is near zero!

Bayes with tied variances e) logistic regression. It is enough to submit these five plots (make sure the axis scales are same so that the plots are comparable). Please do NOT submit code.

# 4 Set-4: Due on 23-Oct-2022 11:59pm

## 4.1 Nearest Neighbour Classification

1. Consider the dataset `https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease`. Partition this set into three equal random parts. Use the first as training set, second as validation set and the final one as the test set. Your goal is to run the k-NN classifier[13] on this data. Use validation set for tuning $k$ among the values 1,3,5,7,9. Partition the input-features for $x$ into three types real $x^r$, integers $x^i$, binary $x^b$. In case the data has categorical features convert them into binary vectorial representations. Use the distance in k-NN as $d$ defined by $d(x,z) \equiv d_1(x^r,z^r) + d_2(x^i,z^i) + d_3(x^b,z^b)$, where $d_1$ is "Minkowski", $d_2$ is "Canberra" and $d3$ is "russellrao". Report the test set error with validation tuned $k$. Now suppose you wish to repeat the same with the generative KDE classifier. How would you go about choosing/designing the smoothness kernel (non-trivial as few input features are real, few others are discrete etc.)? Give as many details as you can and if possible, compare performance of your generative KDE classifier with the test set accuracy with validated k-NN over the same dataset.

# 5 Set-5: Due on 13-Nov-2022 11:59pm

## 5.1 Model Selection

1. Exercise 11.2 in [Shalev-Shwartz and Ben-David(2014)].
2. Exercise 7.10 in [Hastie et al.(2001)Hastie, Tibshirani, and Friedman].

## 5.2 Regularization

1. Repeat Q1 in section 2.2 above with $l2$ regularized logistic regression[14]and SVM[15]. Also, normalize the entire dataset using `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.normalize.html?highlight=normalize#sklearn.preprocessing.normalize` and use only the normalized version for train-test split etc. Tune the regularization hyperparameter, $C = 0.01, 0.1, 1, 10, 100$, using 3-fold cross-validation

[13] `https://scikit-learn.org/stable/modules/neighbors.html#classification`

[14] 'penalty' as 'l2' in `https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression`. $C$ is the hyperparameter to be cross-validated.

[15] set 'loss' to 'hinge' in `https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC`. $C$ is the hyperparameter to be cross-validated.

repeated 5 times. Compare these accuracies with those obtained by you earlier with (unregularized) logistic regression.

2. Repeat Q3 in section 2.1 above with ridge-regression[16] and SVR[17]. Also, normalize the entire dataset using `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.normalize.html?highlight=normalize#sklearn.preprocessing.normalize` and use only the normalized version for train-test split etc. Tune the regularization hyperparameter from the range $0.01, 0.1, 1, 10, 100$, using 3-fold cross-validation repeated 5 times. Compare these explained variances with those obtained by you earlier with (unregularized) linear regression.

### 5.3 Kernel based Models

1. Exercises 16.3, 16.4 in [Shalev-Shwartz and Ben-David(2014)].

2. Exercises 6.1, 6.2 (a)-(c),(e)-(f),(h), 6.3 in `https://cs.nyu.edu/~mohri/mlbook/`.

3. Here your task is to visualize how a kernelized SVM classifier boundary changes with various hyperparameters on `https://archive.ics.uci.edu/ml/datasets/Iris` dataset. Run the kernelized SVM[18] using kernel options: 'linear', 'poly' ('degree' as 2 and ''degree' as 3; other as default), 'rbf' ('gamma' as one of 0.01,0.1,1,10,100). In each case try hyperparameter $C = 0.01, , 0.1, 1, 10, 100$. Submit plots that visualize the data and the classifier along 'sepal length' and 'sepal width' as x,y axis. Organize your plots so that the changes in the classifier as 'gamma' changes or as 'C' changes is well-highlighted.

## 6 Set-6: Due on 30-Nov-2022 11:59pm

### 6.1 Deep learning

1. Here the task is to compare performance of a Feed-Forward Neural Network and SVM with Gaussian kernel trained for classification on the MNIST dataset. Please use PyTorch for training the neural network. You can access the MNIST dataset using torchvision.datasets.MNIST. Using transforms.Normalize, normalize the data to have mean as 0.1307 and standard deviation as 0.3081. Use Cross-Entropy as the loss function with FFNN and SGD for optimization. Train for 20 epochs.

[16] `https://scikit-learn.org/stable/modules/linear_model.html#ridge-regression-and-classification`. $\alpha$ is the hyperparameter to be cross-validated.

[17] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html#sklearn.svm.LinearSVR`. $C$ is the hyperparameter to be cross-validated.

[18] Code at `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC`.

Tune the neural network with minibatch-sizes taken from {10, 100, 1000, 10000}, depth in $\{1, 2\}$. When depth is 1, please take this hidden layer width as 256 and when depth is 2, please take first hidden layer width as 512 and second one's width as 32. And tune the Gaussian SVM with sigma values taken from {0.1, 1, 10, median} where median is the median of Euclidean distances between data points, $C$ in $\{0.01, 0.1, 1, 10, 100\}$. Submit the accuracy obtained on the Test dataset with (tuned and trained) SVM,FFNN. Additionally, submit the plot of training loss over iterations for the neural network code.

2. Exercise 5.18 in [Bishop(2006)].

### 6.2 Representation Learning

1. Exercises 23.1, 23.3, 23.4 in [Shalev-Shwartz and Ben-David(2014)].

2. Exercises 20.4, 20.5 in [Murphy(2022)].

### 6.3 Clustering

1. Generate/simulate the dataset exactly same as third row in the matrix of plots in LINK using a 3-component Gaussian mixture[19]. Using this dataset perform GMM-based clustering (3 components). Visualize the clusters you obtained like in the link. Since you generated the data, you exactly know the means and covariances and the prior. Report the MLE parameters you obtained as well as the true parameters. If they dont match, it is then solely because of the sub-optimality of the EM algorithm.

## References


[Bishop(2006)] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[Hastie et al.(2001)Hastie, Tibshirani, and Friedman] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[Murphy(2012)] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020, 9780262018029.

[Murphy(2022)] Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. URL `probml.ai`.


[19] Use may use code at Link (use # blobs with varied variances). Make sure the 3 Gaussians are sufficiently close by, like in the plots in the link. Else the problem is not challenging.

[Shalev-Shwartz and Ben-David(2014)] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014. ISBN 1107057132, 9781107057135.