Solutions to the exercises for

Machine Learning
A Quantitative Approach

Henry H. Liu

PerfMath

ISBN-13: 978-1986487528
ISBN-10: 1986487520

10 9 8 7 6 5 4 3 2 1
03232019

1 Introduction to Machine Learning

1.1 **Describe the concepts of active potential and refractory period**.

Active potentials are electro-chemical signals fired by neurons when biological cells are stimulated. For example, when you step on a thumbtack, the nearby nerve ends on your sole get stretched suddenly, causing the neuronal membrane's gated sodium channels to become open, which allows Na+ ions to cross the membrane through these channels. The membrane's inside becomes less negative, which causes a discharge process called depolarization. If this depolarization achieves a certain level or threshold, the membrane generates an *action potential*. On the other hand, neurons cannot keep firing action potentials continuously. They have to get relaxed for a period of time before they can fire again. This period is called a refractory period. Figure 1.4 illustrates how an action potential and refractory period look.

1.2 Describe the different roles of the three different types of biological neutrons.

There are three different types of neurons: unipolar sensory neurons, bipolar relay neurons, and multipolar motor neurons. Sensory neurons provide initial input action potentials, relay neurons propagate action potentials, while motor neurons pass action potentials from the nervous system to other tissues of the body. They are similar to artificial neurons at the first hidden layer, middle layers and output layers of an artificial neural network.

1.3 Compare human neurons with artificial neurons. Why can't we build systems that work exactly like human brains?

First of all, on average a human has about 85 billion neurons, while an artificial neutral network can have from a few to a few million neurons. Besides, human neurons are non-generative, i.e., if a biological

neuron dies, that neuron is gone forever. However, artificial neurons never die unless there is a hardware failure or software bug.

Secondly, human neurons work in an electro-chemical process, while artificial neurons are controlled by electro-mechanical signals. This explains why human neurons are equipped with creative, cognitive abilities, while artificial neurons are not.

The above two major differences between human neurons and artificial neurons explain why we can't build artificial neural network systems exactly like human brains, just as airplanes are not built exactly based on how birds fly.

1.4 Compare earlier versions of expert systems with decision tree models invented by Dr. Quinlan. They are all rule-based, but why have expert systems faded away with time but decision tree models persisted, given that both are rule-based?

Expert systems heavily depend on human or *export* knowledge that rules are derived and maintained by humans. However, decision tree models invented by Dr. Quinlan allow rules to be *deduced* by computer algorithms, with given input data. Therefore, the heavy dependency for experts to derive rules based on data is eliminated. This is a clear distinction between earlier expert systems and ML decision tree models.

1.5 What are the pros and cons of linear versus nonlinear machine learning models?

Linear ML models are simpler, less computing resource intensive, while nonlinear ML models are more accurate at the cost of more complex and more computing resource intensive.

1.6 What is the major difference between ML classification and ML clustering?

Both classification and clustering belong to a broader subject of pattern recognition. The major difference between the two is that clustering is unsupervised ML that no labeling or training required, contrary to classification.

1.7 What is the difference between batch learning and online learning?

Batch learning is based on offline data to train a model, while online learning uses real-time incoming data to train a model. Therefore, one is static, while the other is dynamic.

1.8 What are the five ML paradigms as introduced in this chapter?

The five ML paradigms introduced in this chapter include: (1) Rule based learning, (2) Connectivism, (3) Bayesian, (4) Analogy, and (5) Unsupervised learning. Pedro Domingos proposed these five ML paradigms, and §1.3 explains briefly what each of these five ML paradigms is about.

2 Machine Learning Fundamentals Illustrated with Regression

2.1 Try to find a publicly available machine learning dataset and apply an end-to-end procedure similar to the one we used with the fuel economy dataset to come up with your own first linear regression machine learning project. Summarize how you explored the data, pre-processed the data, and what model turned out to be satisfactory in solving the machine learning problem you formulated.

No particular solution. Try a different dataset yourself, e.g., predicting the electrical energy output of a power plant, available from https://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant.

2.2 Explain the concept of under-fitting and over-fitting. How do you determine if the trained model is exhibiting under-fitting or over-fitting behavior? Which one is more likely than the other and why?

Under-fitting refers to large deviations from the data samples, while over-fitting refers to the effects of wiggling through the data samples, which is too good to be true. You can plot the fitted curve against the data used for fitting to identify whether under-fitting or over-fitting is the dominant problem. You can improve under-fitting by increasing model complexity or improve over-fitting by decreasing model complexity.

2.3 Explain the metrics of MSE, RMSE and R^2. How do you calculate and correlate those three metrics? Which metric do you think would measure the performance of a machine learning model more properly?

MSE is just the mean squared errors, while RMSE is just the *root* mean squared error. Neither MSE nor RMSE is normalized, so R^2 is introduced as a normalized entry between 0 and 1 to help better gauge the error of fitting. All these three metrics start with the RSS – the residual sum of errors that measures the error between the predicted and average mean. MSE is the average of RSS, RMSE is the root MSE, while R^2 is 1 minus the ratio of residual sum of squares to the total sum of squares from the observed data. Since $R^2 = 1$ means zero prediction errors, the higher the R^2, the better.

2.4 What's the purpose of cross-validation? List a few most commonly used methods for carrying out cross-validation with a machine learning project.

Cross-validation requires dividing the entire dataset into training dataset, validation dataset and test dataset, instead of just training dataset and test dataset. By doing so, training datasets are better mixed and over-fitting can be minimized by avoiding picking up the noise patterns inherent in the training dataset, thus minimizing generalization error or out-of-sample error. Here, *out-of-sample* means *unseen* data.

Cross-validation schemes include exhaustive and non-exhaustive cross-validations (CVs). The exhaustive CV includes *leave-one-out* and *leave-p-out* with $p > 1$, and the then execute all possible permutations on partitioned data. On the other hand, the non-exhaustive CV includes *k*-fold CV, sampled CV or Monte-Carlo CV, and holdout CV. Please refer to the text for exactly what each of the CV scheme is about.

2.5 What are L1 and L2 norms? How are they used respectively with the Ridge, LASSO and Elastic Net regularizations?

Give two vectors, there are two ways to measure the distance between the two: add the absolute distance in each dimension or add the square of the distance in each dimension and then take the root. The former is known as L1-norm, while the latter is known as the L2-norm. Keep in mind that L2-norm is more sensitive than L1-norm to large-valued outliers.

Ridge and LASSO regularizations are based on L2-norm and L1-norm, respectively, while Elastic Net regularization is based on the mix of two.

2.6 What does a machine learning learning-curve measure? What do we benefit from knowing the learning curve associated with a specific machine learning model?

Learning curve measures the proper amount of training data that minimizes the total error on the validation and test data datasets. When the training dataset is small, the error on the training dataset would be small but large on the unseen data. As the size of training dataset is increased, training error will increase while testing error will decrease. At some point, training error and testing error will converge, which will remain as the training dataset is enlarged. Knowing the learning curve gives us an estimate of the level of *irreducible error*, which can be reduced by improving the precision of training data collected.

2.7 Examine Eqs. (2.26) – (2.28). Explain:

1. **The difference between the concepts of the bias and variance.**
2. **What does the bias-variance trade-off mean exactly?**
3. **Why is it important to understand the implication of the bias-variance trade-off?**
4. **How would you come up with a design of an experiment that would demonstrate the bias-variance trade-off?**
5. **When you conduct your bias-variance trade-off experiment, do you need to keep your test data fixed? Why or why not?**

1. Both bias and variance are relative to average generalization error across multiple datasets. Bias measures the deviation of the average generalization error against the true distribution, while

variance measures the various generalization errors from multiple datasets against the average generalization error.

2. Bias-variance trade-off means that one can adjust the complexity of a model so that the total error as the sum of the variance and bias squared would be decreased.
3. It is important to understand the implication of the bias-variance trade-off as it pertains to the total generalization error, which is one of the most important metrics foe an ML project.
4. When you design an experiment for demonstrating the bias-variance trade-off, it is important that: (1) choose a function to provide a true function to measure against, and (2) choose multiple datasets so that you can measure the average generalization error, bias squared and average variance.
5. When you conduct your bias-variance trade-off experiment, you need to keep your test dataset fixed so that you use the same test dataset to compute the bias squared and variance across multiple training datasets. If you use a different test dataset for every training dataset, then a third has been introduced, in addition to bias and variance.

2.8 In general, from the algorithmic point of view, how would you train a machine learning model and how do you use the trained model to predict on unseen or future events?

This question is related to the NFL (no free lunch) theorem. You should try as many models as possible to find out which model works best for your ML project.

2.9 Check out the implementation of the script named `bias_variance_trade_off.py`. Make sure you understand the overall structure of the script. Trace the following two functions and explain what they accomplish:

- **The function `test_it(degree, alpha)`**
- **The function `compute_bias_variance (h_n_series)`**

These are hands-on coding exercises and readers should complete them themselves.

2.10 Replace the *Ridge* regularization in `bias_variance_trade_off.py` with the *LASSO* regularization and make a few runs to compare with the results presented in the text with the *Ridge* regularization.

These are hands-on coding exercises and readers should complete them themselves.

2.11 Replace the *Ridge* regularization in `bias_variance_trade_off.py` with the *ElasticNet* regularization and make a few runs to compare with the results presented in the text with the *Ridge* regularization.

These are hands-on coding exercises and readers should complete them themselves.

2.12 Some of the programming questions:

1. **We can use Python *list, numpy ndarray* and *pandas Series* to represent 1D data structures. What are the differences among them? How do you convert among them?**
2. **We can use Python *dict, numpy ndarry* and *pandas DataFrame* to represent 2D data structures. What are the differences among them? How do you convert among them?**

3. **How do you add a 1D data structure to a 2D data structure? For example, how do you add a 1D list or *ndarray* to a *DataFrame* object?**
4. **When you create a 3D surface plot like the ones shown in the text, how can you avoid getting the error of *ValueError: shape mismatch: objects cannot be broadcast to a single shape*?**

1. A Python list is not an array as a primitive type. It's more efficient to use a numpy ndarray for numerical computations. You can use `np.asarray(a)`, e.g., to convert a Python list `a` to a numpy array. A pandas Series data structure is a one-dimensional labeled array capable of holding any data type. You can use `pd.Series(a)`, e.g., to convert a Python list `a` to a pandas Series.
2. A Python dict object is a data structure for storing key-value pairs. However, a numpy ndarray is not a key-value data structure, so they not compatible. Similar argument holds true for pandas DataFrame, which is more like a table data structure.
3. For example: `y_ln.append(pd.Series(test_predicted.ravel()), ignore_index=True)` adds a 1D numpy array to a pandas DataFrame object.
4. Make sure proper data structures are passed in.

2.13 Explain why the curves from *m* = 1 and *m* = 2 in Figure 2.22 all look similar as wiggling lines.

The curves are lines with m = 1 as they are supposed to be. The curves with m = 2 are still linear, as a quadratic function does not model an anti-symmetric curve well.

3 Pattern Recognition with Classification

3.1 Explain the full process of a classification machine learning project, based on the examples we have gone through in this chapter.

The full process of a classification ML task includes:

1. With a chosen dataset, explore your dataset so that you become fully familiar with your dataset, such as what features it has, some peculiarities of its data distribution, the dataset size or the volume of samples, etc.
2. Understand how the targets are labeled.
3. Assess whether you need to use stratified data.
4. Perform a few example, exploratory runs with a few models such as default SGD, SGD with `average=True`, SVC, LR, etc.
5. Get a rough estimate of the accuracy score, precision, recall, and F1 scores.
6. Check out the learning curve, i.e., training dataset size to get training error and testing error converged.
7. Explore the effects of adjusting the threshold on the precision, recall and F1 scores. Make trade-offs by adjusting the threshold if desired.
8. Explore other metrics such as the ROC curve and AUC with the model that turns out the best model with your ML project.
9. Explore the confusion matrix with your dataset to find out what samples may easily confuse your ML model. This may provide helpful insight into how the quality of your dataset may be improved over time.

3.2 What are the two most common performance metrics for evaluating a machine learning algorithm or model? How would you make sure both of them have been optimized?

The test accuracy and running time (both training and predicting) are two of the most common and important metrics for evaluating an ML algorithm or model for your ML project. I order to optimizing both of these metrics, you need to try out all best practices scattered throughout the text. Therefore, please take notes of all best practices exhibited throughout the text.

3.3 What's the difference between the two metrics of precision and recall? What controls the trade-off between the two and how and why?

As shown in the definitions of Eqs. (3.2) and (3.3), precision is about the impact of false positives (FPs) or *false alarms* while recall is about the impact of false negatives (FNs) or *misses*. For example, if a person does not have a disease but is diagnosed to have that disease, that's a FP or false alarm. Similarly, if a person has a disease but is diagnosed not having that disease, that is a FN or *miss*. As demonstrated in the text, you can make trade-offs between the two by adjusting *threshold*. A lower threshold would drive down precision with more false alarms while drive up recall with less misses. And a higher threshold would do the opposite. First of all, that's because if the threshold is lower, false positives will be higher, which reduces the number of false negatives. On the other hand, if the threshold is higher, false positives will be lower, which increases false negatives, as a binary classification gives only two outcomes: *true* of *false*.

3.4 What can the AUC score be used for?

AUC scores can be used to cross-compare different models, while a higher AUC score implies a better model.

3.5 Explain how the ensemble methods work in general.

In §3.3.6, we used Random Forests to help demonstrate the concepts of the AUC score. The random forests can be considered a kind of ensemble method, i.e., combining multiple models for an overall score. Ensemble learning is powerful, as it provides more varieties of training datasets, which helps prevent over-fitting significantly.

3.6 Try out binary classification using the MNIST dataset with a different digit than 9 and explain the behaviors of the accuracy score curve and F1 score curve you observe.

You can follow what was demonstrated with the digit '9' in the text with any other digit you want. This would be especially good to help solidify the programming skills you have learnt so far.

3.7 Explore confusion matrices using three different ways as shown in the text.

Similar to the preceding exercise, you can follow what was demonstrated with the digit '9' in the text with any other digit you want. This would be especially good to help solidify the programming skills you have learnt so far.

4 Optimization and Search Illustrated with Logistic Regression

4.1 Is there any other method other than the gradient based ones we introduced in this chapter for optimization problems?

Not publicly known. So far, all optimization problems are based on gradient, as mathematically that's how we can make a judgment on whether a convex or concave has been encountered.

4.2 Under what circumstances should one use batch gradient descent, stochastic gradient descent or mini-batch gradient descent? What are the pros and cons of each method?

Batch gradient descent is impractical as many ML problems have extremely high dimensions, which would take too long to converge. Stochastic gradient descent takes one parameter a time and thus is orders of magnitude faster. The mini-batch gradient is a good compromise between the above two, especially when an ML problem requires traversing the entire parameter space and is memory intensive, e.g., image classification and object identification with convolutional neutral networks to be introduced later. The pros and cons for each method lie with how much of the entire parameter space needs to be traversed and the speed of convergence for the optimization problem in question.

4.3 Try a few runs with the *Iris* dataset using the heavy ball method and explain why it can effectively minimize the zig-zag behavior of a model when getting closer to the optimal point.

The heavy ball method can help minimize the zig-zag behavior of a mode as it is equivalent to some kind of regularization.

4.4 Why could the conjugate gradient method be more efficient than the gradient steepest method?

The conjugate gradient method has a dynamic learning schedule built-in, and thus is more efficient than the gradient steep method that uses a static learning schedule.

4.5 How does the logistic function work as a binary classifier? Are there any other similar functions that can be used as binary classifiers as well?

A logistic function has a property that its values range from 0 to 1 as its argument values vary from $-\infty$ to $+\infty$, which crosses a value of 0.5 when the argument value is zero. Thus, the value of 0.5 of a logistic function provides a critical point that allows us to assign events resulting in negative and positive outcomes as true and false binary events, respectively, which is how a binary classifier works. In fact, a few other functions that bear the similar property to the logistic function, such as the tanh function, can also be used as binary classifiers.

4.6 Why do we use a log function, rather than something else like residual sum of squares, as the cost function for logistic regression?

The inverse of the logistic function ($\mathbf{w}^T\mathbf{x}$) happens to be a log function of a probability quantity, which makes it a suitable candidate for constructing a loss function. It is further transformed into a log loss function that is conformant to the concept of entropy from the information theory, after taking into account the correlations between predicted values and estimated probabilities. On the other hand, residual sum of squares are used for regression ML problems, rather than classification ML problems.

4.7 Describe what the KL divergence is about and how it relates to the softmax regression.

The KL divergence is about measuring the similarity of two similar, correlated distributions. Softmax regression compares the probabilistic distributions of different classes and assigns the largest probability to the class of the sample it belongs to. Therefore, softmax regression falls under the overall umbrella of the KL divergence framework.

4.8 Find two similar distributions and compute their KL divergence. What can you conclude from your computation?

You can find a distribution from your work, and then compare how it has changed from last year to this year, for example. If they are exactly the same, then the KL divergence should be exactly zero. If they are not exactly the same, then the non-zero value should give a measure of deviations from last year and this year.

5 Rule-Based Learning: Decision Trees

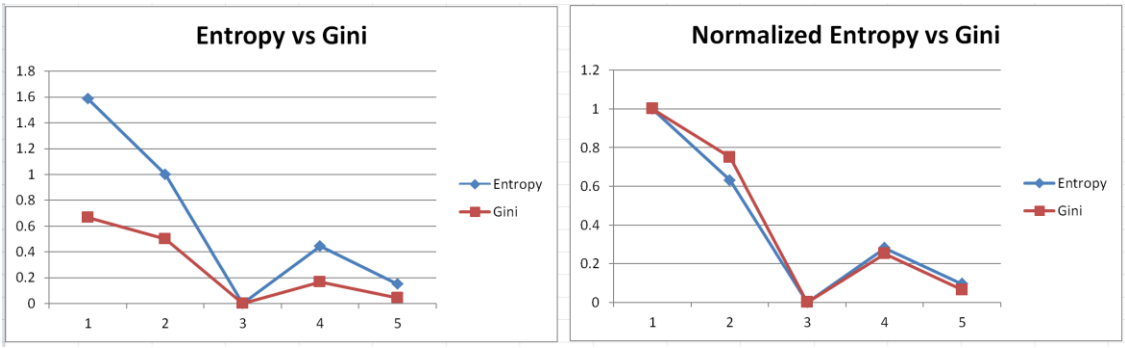
5.1 What are the major differentiates between once-popular expert systems and decision trees, given that they are all rule-based systems conceptually?

The major differences between an expert system and a decision tree include:

- The rules for an expert system are set by human experts, while rules for a decision tree are learnt by machines by deducing from input data.
- An expert system is divided into two subsystems: an inference engine and a knowledge base. The knowledge base represents facts and rules. The inference engine applies the rules to the known facts and deduce new facts. All of these heavily depend on human experts, so expert systems are not truly artificial intelligence. On the other hand, a decision tree is equipped with the capability of learning *autonomously* from externally fed data, so the transit from expert systems to decision trees is a paradigm shift.

5.2 Draw a chart showing both the entropy curve and a Gini-impurity curve on the same *x*-axis. Explain why they seem to have worked equally well.

The following chart is based on data shown in Fig. 5.2 for both entropy and Gini. The left sub-plot used raw data, while the right sub-plot used normalized data for each case. As you see, after normalization, the entropy curve and Gini curve are very similar, indicating that entropy and Gini essentially represent the same probabilistic information.



5.3 Refer to Figure 5.2. Use the last leaf node in each sub-plot to verify the entropy number and Gini number there.

This is meant to be a hands-on exercise, so no solution is provided here.

5.4 How do you determine that your decision tree model has been well trained and will generalize well without incurring over-fitting? If it incurs over-fitting, what measures will you take to overcome it?

Check your trained model with as many unseen datasets as possible. If results vary, most likely it's due to over-fitting. In addition to applying various regularization techniques as discussed in previous chapters, improving the quality of training data may help, e.g., different classes cannot have identical or unknown attribute values.

5.5 How can you transform the *Iris* dataset's sepal attributes to make it easy for applying the decision tree models?

Refer to Fig. 5.5. You can rotate data by ~27 degrees so that *setosa* would be well separated from other two *Iris* classes.

5.6 Since you are already familiar with the regularization techniques in general, find out what regularization options the `sklearn.tree` package provides.

This is meant to be a hands-on exercise, so no solution is provided here.

6 Instance-Based Learning: Support Vector Machines

6.1 Explain the concept of support vectors and why SVMs belong to instance-based learning.

Support vectors are samples lying on the boundary lines to demarcate class boundaries. They are called vectors, as any data point in a n-dimensional space is a vector. SVMs belong to instance-based learning, because the SVM models locate support vectors which are special instances of the entire dataset. Put is another way, training datatset is still relied upon when predicting unseen data, unlike parametric paradigm of machine learning that once training is done, training data are discarded.

6.2 What distinguishes between linear and nonlinear SVMs?

It's the kernel that determines whether an SVM model is a linear or non-linear model. If the kernel function is linear, then it's a linear SVM model. Otherwise, it's a non-linear SVM model, such as Gaussian radial base function (RBF), polynomial, sigmoid, etc.

6.3 What does it mean by hard margin classification versus soft margin classification?

Hard margin classification has no samples crossing the "street" lines, while soft margin classification has.

6.4 What's the purpose of introducing a hinge loss function?

A hinge loss function defines the deviations for the constraints associated with the primal optimization problem. It is made part of the SVM cost function so that it can be minimized to help enforce that the constraints are met as much as possible.

6.5 What is a primal problem and what makes a primal problem a dual problem? What's the motivation to solve a problem as a dual problem for SVMs?

A primal problem is a problem to be solved by solving for proper weight parameters, while a dual problem is another way of solving the same problem by using "kernel tricks," where more kernel functions are available for solving the same problem. This is demonstrated in the text with Eq. (6.10) (primal) and Eq. (6.15) (dual).

6.6 What does it mean by "kernel tricks?"

Kernel trick is also known as kernel substitution, e.g., Bishop, p. 292. If you refer to Eqs. (6.15) and (6.16) in the text, the kernel function depends only on the inner product of the two feature vectors so that it can be replaced or substituted with other kernel functions such as polynomial, sigmoid, or Gaussian radial base function which depends only on the relative distance of the two feature vectors. This is convenient as the inner product or relative distance are easy computations.

6.7 Why are kernel-based SVMs more sensitive to the dataset and more complex, leading to using excessive computing resources?

Kernel-based SVMs are more sensitive to the dataset, e. g., outliers, as its primary job is to determine support vectors that separate once class from others. In addition, kernel methods map the problem to be solved into higher dimensions, which increases complexity and thus excessive computing resources accordingly.

6.8 Run the MNIST benchmarking example on your machine and explain you own observations.

This is meant to be a hands-on exercise, so no solution is provided here.

7 Random Forests and Ensemble Learning

7.1 Summarize what important concepts you have learnt in this chapter about random forests and ensemble learning.

This chapter is intended to help you learn the following concepts about random forests and ensemble learning:

- Random forests that encompass *n* decision trees as estimators, where $n > 1$.
- Ensemble learning that computing the scores with more than one single model. This should be good for combating over-fitting.
- Extra trees that are **ext**remely **ra**ndomized **trees**.
- Instance sampling that samples training data
- Feature sampling that samples in feature space
- adaptive boosting that trains a series of learners with missed instances fed to the next learner in the hope that the next successor can improve upon its predecessor.
- Gradient boosting or gradient boosting regression trees (GBRT) which builds a series of base learners sequentially, with the bias of the combined estimator reduced gradually from one predecessor to another.
- Hard voting that makes decision based on majority voting.
- Soft voting that makes decision based on weighted average from all learners.
- Bagging which allows a sample re-used more than once during instance sampling.
- Pasting which does not allow duplicates during instance sampling.
- Out of bag score which computes generalization error using unused samples from the training dataset.

7.2 Which sampling method allows replacement and which one doesn't?

Bagging allows replacement while pasting doesn't. Here, *replacement* means *duplicates*.

7.3 Between bagging and pasting, which one is more popular or effective?

Based on the experiments presented in Section 7.4.3, bagging seems to be better than pasting.

7.4 Apply GBRT to our fuel economy project and see how much improvement you can get.

This is meant to be a hands-on exercise, so no solution is provided here.

7.5 Write a simple Python script to check the percentage of samples that are not sampled with bootstrap, which should be around 37%. Present your proof or disproof.

This is meant to be a hands-on exercise, so no solution is provided here.

7.6 Check out the two best practices suggested in the summary section with some examples of your own and share your findings with me if you want.

This is meant to be a hands-on exercise, so no solution is provided here.

8 Dimensionality Reduction

8.1 Explain the concepts of eigenvalues, eigenvectors, and matrix diagonalization.

Eigenvalues and eigenvectors describe orthogonal projections for a matrix. Matrix diagonalization is about diagnosing a general matrix by using eigenvectors. However, not all matrices can be diagonalized, A non-diagonalizable matrix is called a *defective* matrix..

8.2 Describe what the singular value decomposition is for and how it leads to PCA.

Singular value decomposition is for finding the most impactful principal components of a data matrix, a metric of score matrix.

8.3 Explain how singular values and eigenvalues correlate with each other.

Singular values are simply the absolute values of the corresponding eigenvalues.

8.4 What does the concept of variance mean for PCA? Why is PCA considered a linear method for dimensionality reduction?

One can consider the variance for PCA as information extent, namely, the higher the more impactful.

8.5 What are the main ideas behind LLE? What merits does LLE have compared with the traditional methods that attempt to solve the same problem?

The main idea of LLE is to model the nonlinear geometry of low-dimensional manifolds. The merits of LLE include:

- Less parameters to deal with
- Does not depend on deterministic annealing

- Achieves global effects by performing local operations.

8.6 Which nonlinear dimensionality reduction method do you prefer, kernel PCA or LLE? State your reasons.

I prefer LLE, especially with those examples shown in color in Fig. 8.5. My reason is that it can be applied to many of my ML problems.

9 Introduction to Artificial Neural Networks

9.1 What is propositional or symbolic logic? Is it a coincidence that the first neural net was conceptually described using propositional logic?

Propositional or symbolic logic deals with true/false propositions. It's not a coincidence that the first neural net was conceptually described using propositional logic, since propositional logic is the zero-th order logic and nothing else can be simpler than that.

9.2 What is the core concept of Hebbian learning? Is it a linear or nonlinear model?

The core concept of Hebbian learning is that neurons that fire together wire together. It is a linear model according to Eq. (9.4), which is understandable that it was proposed in the very early stage of AI.

9.3 What are the profound impacts of Rosenblatt's perceptron model? What is the relationship between his assumption of mutual exclusion of the output units at the output layer with the asynchronous implication of later models such as the Hopfield networks and Boltzmann machines?

Rosenblatt's perceptron model laid the foundation for modern neural networks. Its assumption of mutual exclusion of the output units is consistent with the asynchronous implication of later models such as the Hopfield networks and Boltzmann machines that only one unit is allowed to update its state a time.

9.4 Describe how ANNs have evolved from Rosenblatt's single layer perceptron to the Hopfield networks, Boltzmann machines, restricted Boltzmann machines, and to modern feed-forward, multilayer perceptrons.

Rosenblatt's single layer perceptron is essentially a linear model, while Hopfield networks and Boltzmann machines, including the restricted Boltzmann machines are non-linear machines with hidden layers introduced and simplified connections among neurons.

9.5 What is the key theorem that has made error backpropagation possible?

The chain rule theorem, Eq. (9.28), has made error backpropagation possible.

9.6 What is the complexity of the error backpropagation algorithm?

The complexity of the error backpropagation algorithm is $O(W)$, where W is the weight matrix.

9.7 What physics model inspired the Hopfield network model? And why did that migration inspire so much interest in the AI community?

The Ising spin physics model for explaining ferrormagnetism inspired the Hopfield network model, as Hopfield himself was a physicist.

9.8 What is the thermal equilibrium condition? How does it relate to simulated annealing?

Two physical systems are in thermal equilibrium condition if temperature gradient is zero or no net thermal energy exchanged between them. The process of cooling down slowly from a higher temperature in metallurgy is called *annealing*. In general, simulated annealing is an algorithm for solving combinatorial problems or reaching a global optima with a proper cooling schedule. It has become an actively researched field, with hundreds of papers published each year. In machine learning, it was used in particular for finding the weights of Boltzmann neural networks, which are updated probabilistically, rather than deterministically like Hopfield networks.

9.9 Why did Boltzmann machines have more theoretical merits than practical merits?

It turned out that Boltzmann machines are extremely slow to converge to a thermal equilibrium condition between the visible layer and hidden layer, thus impractical.

9.10 How did restricted Boltzmann machines simplify the generic Boltzmann machines?

Restricted Boltzmann machines have all connections among neurons in the same layer removed, thus much simpler.

9.11 What is contrastive learning divergence and what is its relationship with machine learning divergence?

Contrastive learning divergence is the difference between the initial KL divergence and the KL divergence at the n^{th} iteration, both of which are relative to the eventual distribution at the end of the simulated annealing. The KL divergence for machine learning is just a metric for measuring the similarity of two distributions.

9.12 What problem does Gibbs sampling try to avoid and how does it solve that problem?

Gibbs sampling is used to sample from a conditional distribution when the joint distribution is unknown or hard to sample from. It samples one variable a time with a given conditional probability distribution, which is updated each time with the value of the newly sampled variable.

9.13 List three things you feel you have learnt from this chapter that are interesting.

This is up to each reader.

9.14 Pick a machine learning framework, such as TensorFlow or Caffe or DL4j or any other frameworks you like, and apply multi-layer perceptron to the datasets of fuel economy, MNIST and *Iris* flower we have explored so far.

This is up to each reader.

10 Convolutional Neural Networks

10.1 What's the difference between visual field and receptive field?

Visual field consists of the left and right visual hemifields, while receptive field refers to the region in visual field that responds to external stimuli.

10.2 Describe how human visual systems work.

A human visual system consists of three major parts: visual field, retina and visual cortex. Objects first appear in visual field, then detected by retina, and finally realized by visual cortex.

10.3 What's the difference between convolution and cross-correlation mathematically?

The difference lies in the sign of the shift in the weighting function: convolution has a minus sign like $g(t - \tau)$ while cross-correlation has a plus sign like $g(t + \tau)$.

10.4 What does weight sharing mean?

Weight sharing means a group of neurons on a layer can use or share a same weight set to identify the patterns from an input source, making translation invariance feasible.

10.5. How do you calculate the number of zero-padding units?

Refer to Eq. (10.5).

10.6 What are the differences between convolution layers and pooling layers?

Convolution is about weight fitting while pooling samples input for a different, often down-sized, representation of the input.

10.7 What is your view of a generic CNN architecture?

Come up with your own view and compare with Figure 10.19.

10.8 What have you learnt from those specific CNN architectures we covered?

This is meant for the reader and no solution is provided.

10.9 What's the difference between batch normalization and He initialization?

Batch normalization is about normalizing a batch before it is used, while He initialization is about a form of activation function that is more effective than other known activation functions.

10.10 Pick a deep learning framework and try to implement a CNN of your choice.

This is meant for the reader. No solution is provided. You can search online or refer to Appendix C for some examples.

11 Recurrent Neural Networks

11.1 What's the major difference between a convolutional neural network and a recurrent neural network?

A CNN ML problem is a space domain problem, while an RNN ML problem is a time domain problem.

11.2 What information is considered short-term memory and what information is considered long-term memory?

Input state is considered short-term memory while slowly varying weights are considered long-term memory.

11.3 Describe what dimensions the input and output of an RNN may have.

The input and output of an RNN can be described as a scalar, a vector, or a sequence.

11.4 How would you classify an RNN?

An RNN can be classified into one of the following categories: sequence to sequence, sequence to vector, vector to sequence, and encoder to decoder.

11.5 How would you unfold an RNN in time? What's the purpose of unfolding an RNN?

An RNN can be unfolded in time lags. When an RNN is unfolded, it's easier to build models to deal with its short-term memory and long-term memory characteristics.

11.6 What does it mean by the term *saturation* in the context of an activation function?

It is said that an activation function has reached saturation when its gradient becomes zero.

11.7 When would gradient vanishing or exploding happen? What are the consequences when gradient vanishing or exploding occur? Is gradient vanishing/exploding more severe or less severe with RNNs than with CNNs?

Gradient vanishing happens when gradient approaches zero, while gradient exploding happens when gradient becomes abnormally large. When the former occurs, learning would cease, while when the latter occurs, learning may never converge. Gradient vanishing/exploding is more severe with RNNs than with CNNs, as the longer the time lags, the more challenging to make an RNN work.

11.8 What are kept constant when performing back-propagation through time (BPTT)?

The error flow has to be kept constant when performing back-propagation through time to void gradient vanishing or exploding problem.

11.9 What problem did LSTM solve?

LSTM solved the gradient vanishing/exploding problem with RNN.

11.10 What can one benefit from extending an LSTM unit with a forget-gate?

The introduction of forget-gate allows memory blocks to be reset once their contents become out of date.

11.11 What is the motivation of adding peephole connections with LSTM?

Peepholes are useful for regulating input gate, forget gate and output gate through CEC.

11.12 What is the major difference between GRU and LSTM?

GRU is much simpler in architecture than LSTM. It has only two gates: a reset gate and an update gate. With GRU, when the reset gate is close to 0, the hidden state is forced to ignore the previous hidden state and reset with the current input only. This effectively allows the hidden state to erase any state information that is found to be irrelevant later. On the other hand, the update gate controls how much information from the previous hidden state to carry over to the current hidden state. Because of its simplicity and power, GRU is the most popular RNN model.

11.13 Give three examples of natural language processing.

Speech recognition, automatic summarization and machine translation, as described in the text.

11.14 With the speech recognition example:

1. **What additional extension was introduced to LSTM?**
   A backward layer was introduced with the LSTM model for the speech recognition example introduced.
2. **What's the benefit with a bidirectional neural network? How much improvement was due to bidirectional architecture?**
   A bidirectional model enables both past and future inputs available to the network at the same time. 1% improvement was due to bidirectional architecture.
3. **What regularizations were used? Explain why they helped.**
   Regularization measures include early stopping and adding Gaussian noise to the network weights. Early stopping prevents learning from deteriorating, while adding Gaussian noise to the network weights helps combat overfitting.
4. **What learning rate was used?**
   A learning rate of 10^{-4} was used.
5. **What was the best result achieved?**

The best result was a PER (phoneme error rate) of 17.7%.

6. **Which one is more important, depth or number of cells in a layer?**
   Depth turned out to be more important than the number of cells, as that's what deep learning about after all.

11.15 Pick a deep learning framework and implement one LSTM model or a few.

No solution is provided as this exercise is meant for the reader to complete. You can refer to Appendix D for some example, though.

12 Autoencoders

12.1 Describe how an autoencoder works end-to-end.

An autoencoder consists of an input layer and an output layer, with hidden layers in-between. The dimensions of the input layer and the output layer are the same, while the dimensions of the hidden layers are smaller. It is a symmetric architecture about the midline, so the first half is called an encoder while the second half is called a decoder. Both encoder and decoder use nonlinear functions to encode/decode their inputs. The cost function for an autoencoder is defined by the reconstruction error between the input and output. The objective of training is to minimize the reconstruction error.

12.2 Explain the concept of *coding* or *code*. What can be accomplished by encoding the inputs and then reconstructing the inputs again?

Coding or code is a representation of data, whether it's face data or something else. By encoding the inputs and reconstructing the inputs again, we can come up with very similar but not exactly the same representation of the original data, which has many potential applications.

12.3 Describe three ways for corrupting inputs for denoising autoencoders.

The three ways for corrupting inputs for denoising autoencoders are: adding standard Gaussian noise, masking noise, and adding salt-and-pepper noise.

12.4 Give a complete explanation of the Bayes' theorem. Come up with another example similar to the school girls-boys example to help you solidify what you have learnt about the Bayes' theorem.

The Bayes theorem is a formula for computing posterior probability with given prior probability, marginal likelihood, and marginal probability. The reader should come up with an example similar to the girls-boys example in the text.

12.5 What is a *functional*? What is the essence of the variational method?

A *functional* simply is a function, e.g., a probability function, which is used to define another function, e.g., the *entropy*. The variational method studies the changes caused to the function by changes to the

functional. It is how many machine learning algorithms work, e.g., minimizing a cost with a given functional.

12.6 Give a complete explanation of how variational autoencoders work.

Variational autoencoders are deep generative models based on approximate Bayesian inference and deep neural networks. One representative implementation is by Rezende et al. described in their paper *Stochastic Backpropagation and Approximate Inference in Deep Generative Models*. The common challenges are that both marginal likelihood and posterior density are intractable, as addressed by Kingma and Welling in their paper titled *Auto Encoding Variational Bayes*. Rezende et al. introduced an approximate representation of the posterior over the Gaussian latent variables using a recognition model acting as a stochastic encoder of data. They applied variational principles to optimize the objective function for their generative model, making their autoencoders *variational autoencoders*.

12.7 What is the difference between a generative adversarial net and an adversarial autoencoder?

A GAN uses a generator and a discriminator to generate highly similar data, while an adversarial autoencoder utilizes a GAN to provide a prior to mimic the real data distribution. So the latter depends on the former.

12.8 Pick a deep learning framework and implement one of the autoencoder models introduced in this chapter.

This is meant to be a hands-on exercise. No solution is provided.