

SYSC 4415 Introduction to Machine Learning Fall 2021

Assignment 1

Submission instructions: Please prepare a single Jupyter Notebook with all of your answers. Some answers will only require text while others require text+code+results. Please use the template solution available from the course GitHub repo under "Assignments/Assignment1".

1. Calculate the gradient of the $f(x, y, z) = z^3 + x^2y - y^2 + 3yz$ at $(-2, 3, 1)$. What does this vector represent?
2. For a data science project, you decided to poll your friends to determine the relationship between their need for excess amounts of coffee (greater than 2 cups) versus how many hours of sleep they got the night before during midterm season. One morning you record the number of hours of sleep your friends got on a 1 to 5 scale, since no student gets more than 5 hours of sleep during midterm season. Below are a sample of your results:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 4 | 1 | 3 | 2 | 3 | 2 | 5 | 1 |
|---|---|---|---|---|---|---|---|---|---|

- a. What is the expected value from this sample? Using an unbiased estimator, what are the sample variance and standard deviation?

Oh No! You found out that a research group has already published a similar study. To get an A+ you'll need to compare your results to theirs. They determined the probability mass function, $\Pr(x)$, where x is how many hours of sleep the students got the night before. They also found the probability the students needed excess amounts of coffee given the number of hours they slept, $\Pr(+|x)$. They provide the results in the table below but the study did not report $\Pr(3)$. You need these results for your project! 😞

| x | 1 | 2 | 3 | 4 | 5 |
|------------|------|------|------|------|------|
| $\Pr(x)$ | 0.25 | 0.3 | ?? | 0.10 | 0.15 |
| $\Pr(+ x)$ | 0.75 | 0.55 | 0.35 | 0.2 | 0.1 |

- b. What is $\Pr(3)$?
 - c. Find the expected value and the variance for $\Pr(x)$.
 - d. Find the probability that a student slept 2 hours the night before ($x = 2$), given that they **did not** need excessive amounts of coffee. That is, find $\Pr(2|-)$. *Hint: $\Pr(+)$ can be found by summing over $\Pr(+|x)\Pr(x)$, for all x . $\Pr(-|x)$ can be derived from $\Pr(+|x)$. Bayes' Theorem might be helpful.*
3. Create a python notebook that loads the Kaggle Diabetes Data Set (<https://www.kaggle.com/mathchi/diabetes-data-set>). This dataset has 8 features and 2 classes of diabetes possibility: Outcome: 0=doesn't have diabetes; 1= has diabetes. Hint: look at the notebooks from Tutorials 2 & 3 for example code for achieving the steps below.
 - a. Split the data, using 75% for training and 25% for test. Make sure you use stratified sampling.

- b. Train and test a logistic regression classifier. How accurate is your classifier?
 - c. Repeat part b), but using only the Pregnancies and SkinThickness features from the dataset. Was the classifier accuracy impacted?
 - d. Using the 2-feature classifier from part c), create two subplots using the Pregnancies and SkinThickness features from the data set. See Tutorial 3 Part 2 for similar plots.
 - i. On the first, plot the decision boundary and the training data. Use green for doesn't have diabetes (Outcome==0) and blue for has diabetes (Outcome==1).
 - ii. On the second, plot the decision boundary and the test data. Use the same colours (blue/green), but highlight all misclassified test points (from either class) in red.
4. Linear regression. Download the file "Assig1Q4.csv" from the course GitHub repo under "Assignments/Assignment1". The first column represents the X values, while the second column represents the Y values.
 - a. Plot the data
 - b. We are going to use linear regression to fit a linear and a cubic model to these data. **Without using sklearn.linear_model** (or other linear regression libraries), write your own python code to implement the least squares solution for linear regression. That is:

$$\beta = (X^T X)^{-1} X^T y$$
 - c. Assuming the model $y = mx + b$, use your code to best-fit the parameters m and b to the data. Report your optimal parameter values.

Hints:

 - i. recall that you must create the 'augmented' feature vector X from the given x data (add a column of 1's).
 - ii. look at `numpy.T()`, `numpy.matmul()`, `numpy.dot()`, and `numpy.linalg.inv()`
 - d. Plot your line of best fit on top of the data
 - e. Calculate the mean squared error, as in:

$$MSE(\beta) = \frac{1}{N} \sum_{i=1}^N (y_i - x_i \beta)^2$$
 - f. Assuming the model $y = a^3 + bx^2 + cx + d$, repeat steps 2-4 using this new model (i.e. estimate the optimal values for a,b,c,d; report those estimates; plot the line of best fit; report the MSE).
 - g. Briefly discuss which model would you prefer for these data?
 - h. Why is best-fitting the second (cubic) model still considered linear regression?
5. Create a Jupyter Notebook based on `Tutorial-3_ComparingMultipleClassifiers.ipynb` to use `make_classification` to create a linearly separable dataset, with 2 classes, 2 informative features, **the number of clusters per class to 1**, 1500 samples per class, using a `class_sep=1.7`, and a `random_state` of 5. Generate some random noise of the same shape as your feature data, drawn from **a standard normal distribution** (see `numpy.random`) and a `random_state` of 5.
 - a. Create four datasets: 1) no noise, 2) data + 0.5 * noise, 3) data + 1.0 * noise, and 4) data + 2.0 * noise. For all four datasets, plot the data, labelling each (sub)plot by the degree of noise added (i.e. 0, 0.5, 1.0, and 2.0)
 - b. For each dataset, create training and test data using a 70/30 train/test split
 - c. For each dataset, train and test an SVM classifier with a polynomial kernel with `degree=2`, and `C=1.0`. Report the test score for each. How does prediction accuracy change with noise level?

- d. For a noise level of 0.5, train and test SVM classifiers using the following values for C: {0.001, 0.01, 0.1, 1, 10, 100}. Report the test accuracy for each. How does performance vary with C? Briefly describe what the `c` controls for `sklearn.svc`. *Hint: look at the documentation for `sklearn.svc` rather than the class notes here...*