Assignment #3: Classification & PCA
STA9890
Statistical Learning for Data Mining

Assignment Parameters:

Date Assigned: 2024-03-26
Date Due: 2024-04-09 @ 5:45pm

Submission Mechanism(s):

- Blackboard (strongly preferred)
- Email to instructor: michael.weylandt@baruch.cuny.edu
  Email submissions must be titled *exactly* as `STA9890-S2024-HW2-LASTNAME,FIRSTNAME.pdf`

Question 1: Building a Spam Detector (30 points)

For this problem, you will use the spam data set provided as part of this assignment.

(a) Write a function to perform K-fold cross-validation to select the tuning parameter for ridge logistic regression. You must code this up yourself and cannot use built-in functions (using a built-in function for the base classifier is fine).

(b) Select the optimal tuning parameter using

(1) the minimum CV error rule; and

(2) the one SE rule

for $K = 5$-fold CV. Are the models selected different? Interpret these results and reflect on this.

(c) Perform both $K = 5$ and $K = 10$ fold CV. Does this change the results? Is one of these preferable for this problem?

(d) When reporting the CV error, try out different loss functions:

(1) misclassification error;

(2) binomial deviance error; and

(3) hinge loss error.

Which error function is best for CV and model selection? Why?

(e) Reflect on your results. What have you learned about CV? Which approach to model selection do you think is best for this spam classification example? Why?

(f) Use a model selection procedure to select tuning parameters for each of the following classifiers: Linear SVM, Gaussian Kernel SVM, and Polynomial Kernel SVM.

(g) Report the accuracy (model assessment) of each classifier for this spam data set. Which one is best? Why? Interpret and reflect on your results.

(h) Discuss why your model selection and assessment procedures are correct and justify any decisions you made.

Note: For parts (f-h), you may use any built-in functions. The question is purposefully vague as it is up to you to design and implement a correct model selection and model assessment scheme to decide which type of SVM classifier is best for building a spam filter.

Question 2: Exploratory Analysis & Unsupervised Learning (20 points)

Use PCA, NMF, and ICA to find patterns, reduce the dimension, and visualize the data. Please download the *Digits Data* from the ESL webpage (you used this data in HW1).

(a) Visualize results from the 3 methods. How would you visualize patterns among the samples? Among the features? Show these graphics, explain them, and interpret the results. What do these reveal? Do you find anything interesting?

(b) How much variance is explained by each PC? What would be a good number of PC factors to retain for this data? Explain.

(c) How do the results of ICA and NMF change when you take $r = 10, 20, 50, 250$ factors? Is there a way that you could decide how many factors to retain in a data-driven manner? Explain.

(d) Is there a quantitative and objective way to that you can determine which is the best pattern recognition technique for this data set? How? Explain and implement your procedure.

Question 3: Properties and Applications of the SVD (20 points)

- Prove: If $\boldsymbol{X}$ is a matrix, the left singular vectors of $\boldsymbol{X}$ are the eigenvectors of $\boldsymbol{X}\boldsymbol{X}^\top$ and the right singular vectors are the eigenvectors of $\boldsymbol{X}^\top\boldsymbol{X}$ (5 points)
- Compute the ridge regression solution in terms of the SVD of $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^\top$. Describe how the entire set of ridge solutions can be computed efficiently once the SVD of $\boldsymbol{X}$ is pre-computed. (5 points)
- *Principal Components Regression* (PCR) refers to the process of:

  1. First, projecting the data matrix $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ onto a small number of principal components to get a reduced data matrix $\tilde{\boldsymbol{X}} \in \mathbb{R}^{n\times k}$
  2. Fitting a linear model (*e.g.* OLS) on $(\tilde{\boldsymbol{X}}, \boldsymbol{y})$.

  This method typically has better performance than a pure linear model, with the PCA pre-processing essentially acting as a form of regularization.

  Implement PCR on the gene marker data set from the previous homework and compare its predictive performance to OLS, ridge, and lasso. Use K-fold CV to select the optimal tuning parameters for each method. (5 points)
- Using the SVD of $\boldsymbol{X}$, derive a closed-form solution for $\hat{\boldsymbol{\beta}}_{\text{PCR}}$: compare and contrast this with your SVD-ridge solution. (5 points)

Question 4: Supervised Dimension Reduction Methods (10 points)

PCA is an *unsupervised* dimension reduction method, but supervised analogues exist. The most famous of these are methods like *Canonical Correlation Analysis*[1] Unlike our typical "decompose $\boldsymbol{X}$" story, CCA/PLS finds a paired decomposition of two matrices $\boldsymbol{X}, \boldsymbol{Y}$ where the rows correspond to the same observational unit. Unlike supervised learning, here $\boldsymbol{Y}$ is a *matrix* and we don't have a single all-important scalar response. We can fit these methods by finding the SVD of $\boldsymbol{X}^\top\boldsymbol{Y}$: the resulting singular vector (pairs) capture the elements of $\boldsymbol{X}$ that best predict a combination of elements of $\boldsymbol{Y}$. In this question, you will fit PLS/CCA on the `palmerpenguins` data.

- Create $\boldsymbol{X}, \boldsymbol{Y}$ as follows:

```
library(palmerpenguins)
Y <- model.matrix(~0+ species, data=na.omit(penguins))
X <- as.matrix(na.omit(penguins)[,c(3, 4, 5, 6)])
```

- Compute the first and second pair of singular vectors of the cross product matrix $\boldsymbol{X}^\top\boldsymbol{Y}$.
- Based on the first left singular vector, what is the most important variable to predict species?
- Based on the second pair of singular vectors, what body feature is most useful for predicting which species?
- How do your results compare to a PCA analysis of this data? `https://allisonhorst.github.io/palmerpenguins/articles/pca.html`

[1] CCA is very closely related to another method called *Partial Least Squares* (PLS)l here we'll treat the two interchangeably.

You should not expect classifiers built using CCA/PLS output to do *as well as* pure classifiers, but they can add useful insights.

Question 5: (Regularized) Power Methods for PCA (20 points)

In class, we discussed how the *singular value decomposition* (SVD) can be used to perform PCA. In this question, we will explore a classical method for computing the SVD and explore how it can be adapted to non-classical PCA variants.

Our starting point is the *power method* for computing the leading eigenvector of a positive definite matrix:

Algorithm 1 Power Method for Matrix Eigenvectors

Inputs: $\boldsymbol{\Sigma} \in \mathbb{R}^{p \times p}_{\succ 0}$

Initialize: $\boldsymbol{v}^{(0)}$ to be a random unit vector.

- Sample p standard normal random variables to create $\tilde{\boldsymbol{v}}^{(0)}$
- Normalize $\boldsymbol{v}^{(0)} = \tilde{\boldsymbol{v}}^{(0)} / \|\tilde{\boldsymbol{v}}^{(0)}\|_2$

Repeat Until Convergence:

- $\tilde{\boldsymbol{v}}^{(k+1)} = \boldsymbol{\Sigma} \boldsymbol{v}^{(k)}$
- $\boldsymbol{v}^{(k+1)} = \tilde{\boldsymbol{v}}^{(k+1)} / \|\tilde{\boldsymbol{v}}^{(k+1)}\|_2$
- Set $k = k + 1$

Return:

- Estimated Eigenvector: $\boldsymbol{v}^{(k)}$
- Estimated Eigenvalue: $\hat{\lambda} = \|\boldsymbol{\Sigma} \boldsymbol{v}^{(k)}\|_2 / \|\boldsymbol{v}^{(k)}\|_2$

- Using the spam data from the previous problem, compute the first principal component "by hand:" (5 points)
  - Center the data matrix
  - Compute the covariance matrix using the centered data matrix (you *may not* use the built-in `cov` function here).
  - Implement Algorithm 1 to compute the first principal component. $\boldsymbol{\Sigma}$ should be your estimated covariance.

  Compare your result to what you could obtain using `prcomp`. If it differs, explain why.
- The eigenvector power matrix can be modified to compute the singular vectors instead: Implement Algorithm

Algorithm 2 Power Method for Matrix Singular Vectors

Inputs: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$

Initialize: $\boldsymbol{u}^{(0)}, \boldsymbol{v}^{(0)}$ to be random unit vectors of length n, p respectively.
(You can use the same normalized random Gaussian approach as Algorithm 1.)

Repeat Until Convergence:

- $\tilde{\boldsymbol{v}}^{(k+1)} = \boldsymbol{X}^\top \boldsymbol{u}^{(k)}$
- $\boldsymbol{v}^{(k+1)} = \tilde{\boldsymbol{v}}^{(k+1)} / \|\tilde{\boldsymbol{v}}^{(k+1)}\|_2$
- $\tilde{\boldsymbol{u}}^{(k+1)} = \boldsymbol{X} \boldsymbol{v}^{(k+1)}$
- $\boldsymbol{u}^{(k+1)} = \tilde{\boldsymbol{u}}^{(k+1)} / \|\tilde{\boldsymbol{u}}^{(k+1)}\|_2$
- Set $k = k + 1$

Return:

- Estimated Left Singular Vector: $\boldsymbol{u}^{(k)}$
- Estimated Right Singular Vector: $\boldsymbol{v}^{(k)}$
- Estimated Singular Value: $\hat{d} = (\boldsymbol{u}^{(k)})^\top \boldsymbol{X} \boldsymbol{v}^{(k)}$

2 and apply it to the spam data. Compare your results to the output of calling `svd` directly. (5 points)

- We can modify the classical power method to introduce regularization ideas like sparsity into PCA. Specifically, if we want k-sparse PCA, we can use something like the Algorithm 3. Here, we modify the power method to add a truncation step under which all but the top K largest elements of a vector a set to zero.

Implement Algorithm 3 and apply it to the spam data set. Which features does sparse PCA select as the most important? Is this consistent for different values of K? (10 points)

Algorithm 3 Power Method for K-Sparse Matrix Singular Vectors

---

Inputs: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$

Initialize: $\boldsymbol{u}^{(0)}, \boldsymbol{v}^{(0)}$ to be random unit vectors of length n, p respectively.
(You can use the same normalized random Gaussian approach as Algorithm 1.)

Repeat Until Convergence:

- $\tilde{\boldsymbol{v}}^{(k+1)} = \boldsymbol{X}^\top \boldsymbol{u}^{(k)}$
- $\hat{\boldsymbol{v}}^{(k+1)} = \mathsf{TopK}(\tilde{\boldsymbol{v}}^{(k+1)}, K)$
- $\boldsymbol{v}^{(k+1)} = \tilde{\boldsymbol{v}}^{(k+1)} / \|\tilde{\boldsymbol{v}}^{(k+1)}\|_2$
- $\tilde{\boldsymbol{u}}^{(k+1)} = \boldsymbol{X}\boldsymbol{v}^{(k+1)}$
- $\boldsymbol{u}^{(k+1)} = \tilde{\boldsymbol{u}}^{(k+1)} / \|\tilde{\boldsymbol{u}}^{(k+1)}\|_2$
- Set $k = k + 1$

Return:

- Estimated Left Singular Vector: $\boldsymbol{u}^{(k)}$
- Estimated Right Singular Vector: $\boldsymbol{v}^{(k)}$
- Estimated Singular Value: $\hat{d} = (\boldsymbol{u}^{(k)})^\top \boldsymbol{X}\boldsymbol{v}^{(k)}$
