Midterm Lab 5 Extension

Anonymous submission

Iris Dataset Classification

Comparison of PCA and LDA 2D projection.

This uses the iris dataset to train and plot the PCA and LDA algorithms. The PCA finds the attributes with most variance between data. The LDA finds the attributes with the most variance between different classes. They both modeled based on two components and plotted graphs with different colors for different target names

Like many of the other uses of the iris dataset this uses the data to train the algorithms, and uses two attributes to reduce the dimensions. They also have plots, but unlike many of the other uses, there is no background color map or decision boundaries.

Plot the decision surface of decision trees trained on Iris data set

Take different combinations of 2 attributes to train with data with the decision tree classifier. Then plot the different decision tree boundaries with the training data points for the six pairs of attributes. Then show the structure of a single decision tree that was trained on all of the features used together.

Similar to the other uses of the iris dataset, this trains with the classifier, and uses two attributes to reduce dimensions. Though this specifically uses six different pairs of attributes and plots graphs for each pair. The graphs all have decision boundaries which some others don't have. This also plots the tree uses every feature.

Plot the Decision surfaces of ensemble trees on the iris dataset

Using decision tree, random forest, extra tree, and AdaBoost, and comparing them. The ensemble methods used 30 estimators. Using different 3 different pairs of attributes to train. Plots the graphs and gives scores for each classifier.

This trains similarly to other uses, but this ensemble methods uses 4 different methods. It also reduces to two attributes like most of the others. This uses estimators, and most others don't. This also plots the graphs with training points and decision boundaries like others.

Understanding the decision tree structure

Use the iris dataset and train test split for decision tree classifier. They use arrays to move through the tree structure and they compute the depth of each node and whether or not the node is a leaf. They can get the decision path for different samples and determine common nodes samples go for a group of samples.

This uses a train test split which most others don't do. This tries to compute various properties of decision trees, while most of the other uses are for training based on data and visualizing the results.

Plot multiclass SGD on the iris dataset

Only use two attributes to reduce dimensions. Plots decision surfaces and boundaries along with training datapoints. Also plots hyperplanes for three one against all classifiers.

Like many others this only uses two attributes to reduce the dimensions. It also uses the data for training. This also visualizes through a plot with training points and decision boundaries. Unlike the others this plots three hyperplanes for one against all classifiers.

Plot the decision boundaries of a voting classifier

The voting classifier uses decision tree classifier k neighbors classifier and svc classifier with weights of 2 for decision tree and svc and 1 for k neighbors. So the probabilities for tree and svc wil have double the weight of k neighbors when averaged. Only two attributes are used to train the data. The decision tree, k neighbors and svc are all plotted along with the voting classifier plot.

Similarly the the other uses this only uses two attributes to reduce dimensions and uses data for training. It also has plots with decision boundaries and training data. This uses multiple methods and has a plot for each different method, and unlike others it has a plot that combines the different methods with different weights.

Regularization of L1 path - Logistic Regression

Trains a logistic regression model on classification problem from iris dataset. 4 coefficients of model are gathered as a path. That path is then plotted.

This similarly to other uses visualizes with a plot. Unlike others this doesn't reduce the dimensions. It also plots for a path, instead of decision boundaries and training points.

Nearest Neighbors Classification

Only uses two attributes to reduce the dimensonality. Trains data and plots training points along with decision boundaries in colormap. They use two different weights with one uniform, and the other with a distance weight. Both are trained for 15 neighbors.

This reduces to 2 attributes like many others, and uses data for training. Like others this plots decision boundaries with training points. Unlike others this has different plots for different weights, so that it shows different types of nearest neighbor classifications. It also takes a parameter of 15 neighbors, and not many of the others take use parameters.

SVM Anova

Runs feature selection before running SVC to improve classification scores. Creates a pipeline that creates feature selection transform, scaler, and instance of SVM and combines for an estimator. Plots cross validation score against percentiles.

Unlike other uses this is trying to improve classification scores before running SVC. So it is used to improve the model before training the the data. This has a plot to visualize like most of the other uses. But this plot is for the cross validation score and percentiles of features.

Diabetes Dataset Regression

Linear Regression

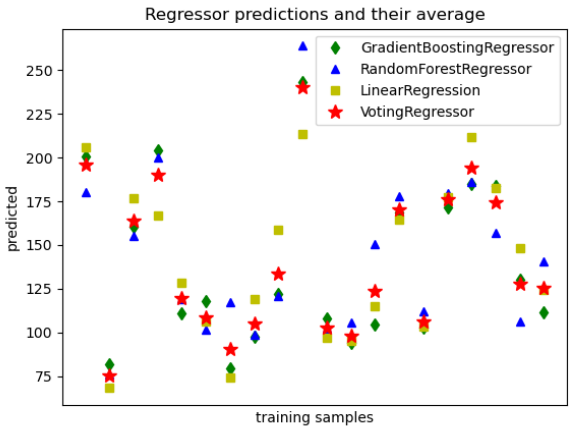
For this example, the model takes in one feature to to fit a linear model based on the data. It splits the data into training and testing sets. The model is trained using the training set and the testing set is used to make predictions about the data. The linear model is created by minimized the sum of the squares between the observed and predicted data. This prints the coefficients for the model and the mean squared error. The testing data is plotted in a graph for visualization along with a line for the linear regression model. The regression for the data trained and testing in this example based on the coefficient would be $y = 938.24x$

Gradient Boosting Regression

In this example, the outputs of multiple predictive models were used as an input to an ensemble method. The model uses a number of boosting stages to train the model and in this example, 90% of the data was used to train while the remaining 10% was used to test the model. From all of the weaker predictive models, it was found that the feature "BMI" and "s5" were the strongest determining features correlated to diabetes. This regression model is helpful to find a common pattern between multiple different models that might have mixed outcomes.

Voting Regressors

In this model, three other models were used as inputs: gradient boosting model, a random forest model, and a linear regression model. The voting regressor model essentially finds a coefficient, or a weight, to each of the three models. In this case, there were 10 features to be "voted" on, or to decide how heavily each of these 10 features were correlated to diabetes. Each of the three models have their own opinions on how much each of the features are correlated to having diabetes and based on these decisions, the voting regressor will decide how much each model will contribute to a final score for each feature. In this model, the final decision was plotted in a red star, which looks to be roughly an average of the three decisions.



LASSO Regression

LASSO, standing for Least Average Shrinkage and Selection Operator, is a regularization model used to make more accurate predictions. LASSO is similar to a linear regression is that it shrinks data, or is a rough average of the data plots, but the weights in the equations will be low, sometimes even 0. Linear regression is not penalized by having large weight values as the regression model is based on a difference of squares while if the absolute value of the weights in the regression model of LASSO models are high, the model will suffer as it is based on a difference of absolute value.

Feature Selection

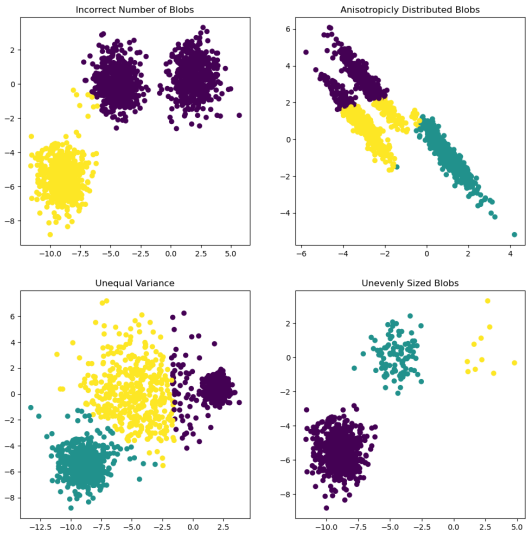
In this example, a RidgeCV function was used to calculate the weights of the features in determining whether or not a person has diabetes or not. For each of the 10 features, a coefficient or weight was assigned—the higher the weight, the more likely that the feature could be attributed to diabetes in the person. If we wanted to select to two most important determining features, there are two methods: SelectFromModel and SequentialFeatureSelection. SelectFromModel essentially calculates the top two features based on the coefficients of a single model (this case being RidgeCV). SequentialFeatureSelection (SFS) performs the same task but also cross-validates its prediction with other models. For instance, the diabetes data set was analyzed using the RidgeCV model. SFS would make a prediction of the features on the RidgeCV model and cross-check that prediction with other models besides from RidgeCV (such as linear regression or LASSO regression).

Advanced Plotting with partial dependence

In this example, decision tree and multi layer perceptron regressors are used to train data to plot partial dependence plots. They plot based on two features, age and bmi. They can show 4 plots separately for each model and feature, or the can show 2 plots where they combine the models for each feature onto one plot to compare.

Unsupervised Learning and Clustering

Our understanding of unsupervised learning is training some data and organizing the data in a way that reveals an implicit pattern. The algorithm then interprets the data based off some previous knowledge and tries to identify the pattern. So the algorithm takes unlabeled data, and finds patterns within the data to organize the data into different groups. A common way the algorithm can categorize data is with clustering, or identifying different groups of data. For instance, if we have the following plot graphs, we can see that the program identifies each unique group or cluster as a different color. The program itself doesn't really know that these groups necessarily mean anything, but is capable enough to identify that they are inherently unique.



Whether or not the program decides to cluster plots as a certain way (for instance, the upper right graph might seem unintuitively divided) is up to the algorithm and how fitted it may be.

K-means Clustering

In k-means clustering, the user defines how many clusters there should be and the algorithm tries to identify the clusters that exist. The algorithm essentially assigns a "center" to what it thinks is a cluster and begins to assign new centers based on the values around the old center. This can be done with a mean or a minimum distance function. When it finds that the difference between the old and the new centers are small enough, it will state that the closest points to the final center will be included in the cluster around the center.

Affinity Propagation

For affinity propagation, pairs of samples send messages to each other until there is convergence. Dataset is described by a small number of points deemed most representative of the other samples. Messages are used to find of suitably a point of data may be to represent another point of data. This happens until the final representatives are chosen and each representative are the focal point of the each cluster. It is very complex and best used for medium sized datasets.

DBScan

In DBScan clusters are seen as areas of high density separated by areas of low density. It uses parameters, min samples and eps and higher min samples or lower eps indicates higher density for clusters. A sample where there exist min sample other samples within a distance eps is a core sample, and the other samples are neighbors. A cluster is found by finding a core sample, and then finding the core samples in the neighbor, and recursively finding core samples until all the core samples are found. Then the core samples are combined with the other samples that aren't core samples but are still neighbors of a core sample to form a set that is a cluster.

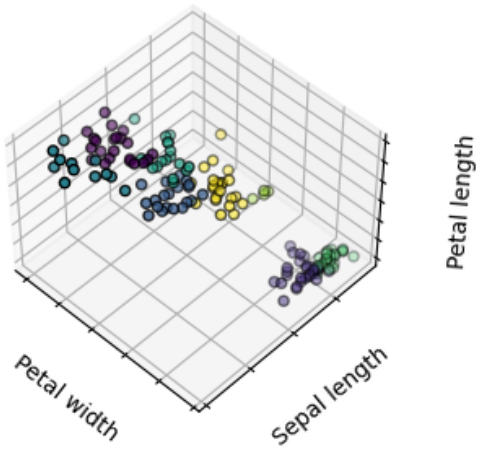
Signal Decomposition

Basically, if you have a signal with many different other signals inside, we want to be able to break it down into its components. For instance, if you listened to a song, you can distinctly hear the different components of the song: the vocals, the guitar, the drums, the piano, etc. This idea can be extended to many different concepts such as identifying the features of a face or the features of a flower.

Favorite Clustering Iris Data Set Examples

K-Means Clustering

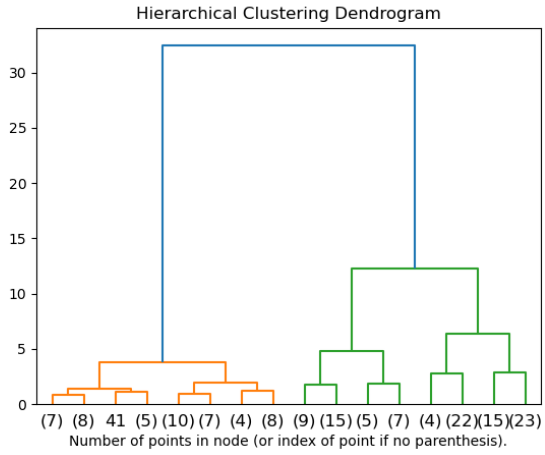
In this example, the Iris data set was classified into 3 clusters using a K-means clustering algorithm. What was interesting about this example was the difference between allowing the center to shift only once compared to multiple times.



The upper left hand was the clustering example where the algorithm was only allowed to shift each center only once, which resulted in a very vague separation between the clusters signified by the lack of separation in color. As the number of k increases or the number of centroids increase, the more finely defined the clusters are. After a certain point, the differences between the cluster separations is so little it does not make sense to add any more centers or define any clusters. This ”turning point” is called the elbow and can be found using an ”elbow method”. The elbow point for the Iris data set is naturally around 3 clusters as there are 3 distinct clusters.

Hierarchical Clustering Dendrogram

This example highlights how an algorithm can determine a number of specified clusters based on the hierarchy of the ”fork”. If we draw a horizontal line and start moving it from top to bottom, we can see how each of the branches can split into different groups. As we move down the graph, we can increase the number of sub-divisions and see exactly which points are contained in each sub-division.



Affinity Propagation

One of our favorite examples for the iris dataset was the affinity propagation. From the example Kaggle dataset, the model was found to be pretty accurate for the iris dataset. They estimated the number of clusters to be 3, and they took many measurements about accuracy and performance metrics. The accuracy found which percentage of the labels were correct, and this model had an accuracy of .9 which is a strong score. They also used multiple metrics like homogeneity, which measures how much each cluster has only datapoints of one class. The model had a solid homogeneity score of 0.77.

Literature Review

Maulud and Abdulazeez (2020) reviewed the results of multiple papers on regression from 2016 to 2020. The different types of regression for the papers were single linear regression, multiple linear regression, and polynomial regression. The multiple linear regression papers were found to have the best accuracy.

Roopa and Asha (2019) built a model to improve accuracy for the diabetes dataset. The model uses principal component analysis to extract features of the data and multiple linear regression to fit the model. The model was found to have 82.1 percent accuracy.

Zhang, Deng, and Su (2016) used an algorithm that aims to find a linear model using missing data. They are looking at an incomplete dataset on diabetes because they are studying linear models on glycosolated hemoglobin, which is a measure of diabetes. They compare the algorithms performance with other methods and find that the algorithm performs better.

Alhamzawi and Ali (2018) used Bayesian approaches to provide consistent estimates for regression coefficients in adaptive LASSO regression. They compare the performance with existing Bayesian and non-Bayesian approaches and they find that their new approach performs well in comparison.

Sharma (2018) proposed a new variation of SGD that tries to overcome SGD’s bias toward random selection of a data instance. It tries to do this by using greedy selection of consistent data instances. It was tested along other popular variations of SGD one many datasets including diabetes and was found to have better convergence and accuracy.

Zhang Duchi and Wainwright (2013) propose a different method divide and conquer method for finding kernel ridge regression. This method partitions the dataset into subsets of equal size and computes a kernel ridge regression estimator for each subset and averages them. This reduces the computational time, and they argue that the statistical optimally is still retained.

Fan, Chen, Li, and Zhu (2015) compared Lasso, adaptive lasso, ridge, and elastic net regression on the diabetes dataset. They look for variable selection and model prediction to evaluate the regression models. Akaike information criterion, bayes information criterion, and cross validation are used to evaluate variable selection. They find that lasso has the best variable selection, but has less accurate predictions, and elastic net finds the most accurate predictions.

Jayanthi, Babu, and Rao (2017) describe various predicative models for diabetes. They discuss models using neural networks, k neighbors, decision trees, support vector machines, and other classification models. There are also regression models like multiple linear, ridge, lasso, and elastic net. Hybrid models are also discussed, and they find the hybrid models to be more accurate. They find that elastic net would be the most useful for diabetes.

Naz and Ahuja (2020) use machine learning algorithms to predict diabetes. They use atrificial nueral network, naive bayes, decision tree, and deep learning and finds they all have accuracies of at least 90 percent. The highest accuracy is deep learning which has an accuracy of 98.07 percent.

Kumar and Prenavi (2017) look into the use of machine learning algorithms for predicting diabetes. They look at classification algorithms and test for accuracy. They test random forest, support vector machine, k nearest neighbors, classification and regression tree, and linear discriminant

analysis. The random forest algorithm was found to have the best accuracy results.

References

Maulud, D., & Abdulazeez, A. M. (2020). A Review on Linear Regression Comprehensive in Machine Learning. Journal of Applied Science and Technology Trends, 1(4), 140-147. https://doi.org/10.38094/jastt1457

Roopa, H., & Asha, T. (2019). A Linear Model Based on Principal Component Analysis for Disease Prediction. IEEE Access, 7, 105314-105318.

X. Zhang, J. Deng & R. Su, (2016) "The EM algorithm for a linear regression model with application to a diabetes data," International Conference on Progress in Informatics and Computing (PIC), 2016, pp. 114-118, doi: 10.1109/PIC.2016.7949477.

Alhamzawi, R., & Ali, H. T. M. (2018, June 18). The Bayesian Adaptive LASSO Regression. Mathematical Biosciences. https://doi.org/10.1016/j.mbs.2018.06.004

Sharma, A. (2018, October 12). Guided stochastic gradient descent algorithm for inconsistent datasets. *Applied Soft Computing*. https://doi.org/10.1016/j.asoc.2018.09.038

Zhang, Y., Duchi, J. & Wainwright, M.. (2013). Divide and Conquer Kernel Ridge Regression. *Proceedings of the 26th Annual Conference on Learning Theory*, in *Proceedings of Machine Learning Research* 30:592-617 Available from https://proceedings.mlr.press/v30/Zhang13.html.

Lei Fan, Shuai Chen, Qun Li & Zhouli Zhu (2015) "Variable selection and model prediction based on Lasso, adaptive lasso and elastic net," 4th International Conference on Computer Science and Network Technology (ICCSNT), 2015, pp. 579-583, doi: 10.1109/ICCSNT.2015.7490813.

Jayanthi, N., Babu, B.V. & Rao, N.S. (2017). Survey on clinical prediction models for diabetes prediction. J Big Data 4, 26. https://doi.org/10.1186/s40537-017-0082-7

Naz, H., Ahuja, S. (2020). Deep learning approach for diabetes prediction using PIMA Indian dataset. J Diabetes Metab Disord 19, 391–403. https://doi.org/10.1007/s40200-020-00520-5

P. S. Kumar & S. Pranavi, (2017) "Performance analysis of machine learning algorithms on diabetes dataset using big data analytics," 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS), 2017, pp. 508-513, doi: 10.1109/ICTUS.2017.8286062