Modeling Madelon dataset

Goals in this project:

With the Madelon dataset, my goal is to create a model pipeline that can reduce the dimensionality of the dataset by perform feature selection and also accurately predict the 'target' label. This dataset is synthetic by nature and is a 2-class classification problem. The features are continuous and are not described which makes it hard to interpret the results.

Data:

I am using a Madelon dataset generated by Joshua Cook which has around 200,000 instances and 1000 features. I used psycopg2 SQL connection through Pandas to get access to his postgres database and read the data into csv file to store. I read 20000 instances of the database at a time to read into csv file as I was unable to read the entire dataset at once due to the limitation of the T2 micro instance in AWS. I am using 2 samples of 20000 instances of the entire dataset to run my model pipeline

Benchmark:

The target for this dataset are binary classes (1, -1), hence I fit naive LogisticRegression, KNeighborClassifier, DecisionTreeClassifier and Support Vector Classifier to get benchmark score on the the datasets. As there is no prior information about the data features and the relationships between them I am using model like LogisticRegression and SVC for linear data and KNeighbors and DecisionTreeClassifiers for nonlinear data.

I used 1% of the data (2000 instances) to perform benchmarking.

	LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1, penalty='l2', random_state=None, solver='liblinear', tol=0.0001, verbose=0, warm_start=False)	KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=5, p=2, weights='uniform')	DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best')	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf', max_iter=-1, probability=True, random_state=None, shrinking=True, tol=0.001, verbose=False)
0	0.533724	0.614236	0.703796	0.651159

Table.1 Benchmark score of 1% Madelon data using 4 different classifier

Performance Metric:

I chose to use area under the curve for Precision-Recall curve to measure the classifier's performance, as I have no real indication of what the features mean.
I separated the target column from predictors of dataset.

Feature Selection:

Madelon dataset has 1000 predictor and many of which might be redundant. It becomes essential to reduce number of predictors as the model might fit poorly on redundant predictor. I used 4 different techniques to search for predictors that are informative and not redundant.

1) iterative method which checks each feature of the dataset against rest of the dataset and computed the R2 score to determine if that particular feature can predict the dataset. I use DecisionTreeClassifier as an estimator for this method. In this method I collect the R2 score for each feature, and pick top features with highest R2 score. R2 score is a measure of the percentage of data can be predicted/explained by your model, hence a higher R2 score is chosen.

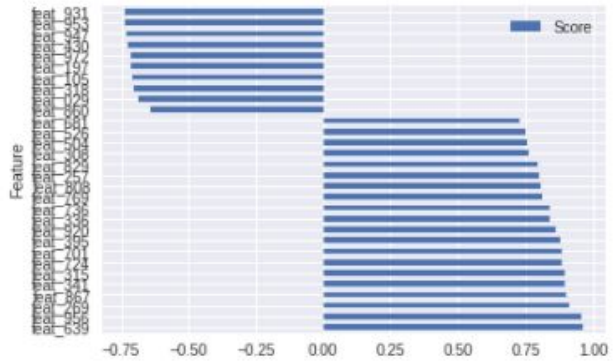

Fig .1 Plot of feature R2 score on X-axis and feature on Y- axis.

2) covariance matrix to measure the covariance between each features and I selected the top features with highest covariance.

3) SelectKBest and set k=100 features to select. SelectKBest uses F-score as metric to choose the features which is a measure of accuracy and considers both precision and recall.

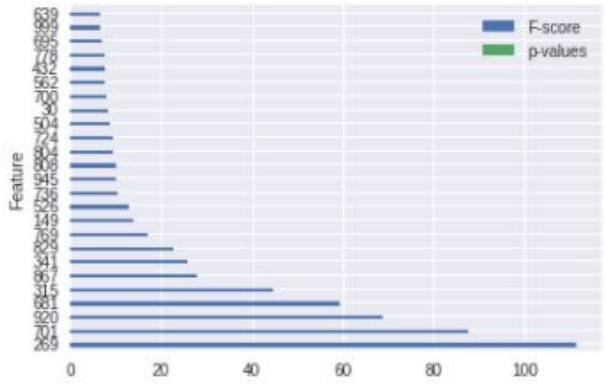

Fig. 2 Plot of F-score on X-axis and feature on Y-axis after performing SelectKBest with k =100

4) RandomForestClassifier inside Gridsearch to get feature importances.

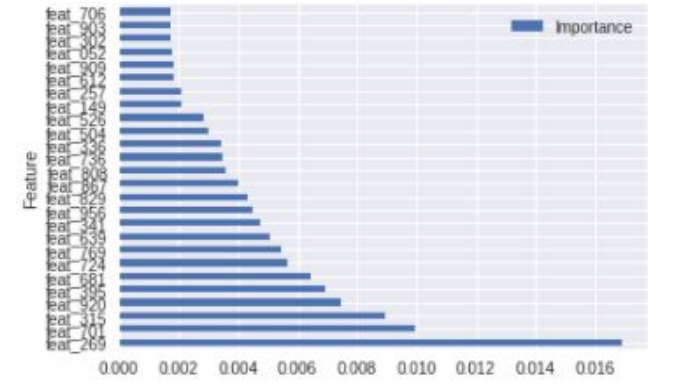

Fig. 3 Plot of feature importance on X-axis and feature on Y-axis after performing RandomForestClassifier on Madelon dataset.

For all these methods I sorted the features scores or importance, plotted a bar graph to visually inspect features that are informative.

I came up with a list of 20 features which appeared in at least 2 of the above mentioned methods. I justify that these are the most important features in the dataset as they are selected by many of these methods.

sel_feature= [257,269,308,315,336,341,395,504,526,639,681,701,724,736,769,808,829,867,920,956]

EDA on the 20 features selected:

After selecting 20 features, and before performing GridSearch I once again score the 4 classifiers to observe improvement in PR_auc score. I used 20000 instances with 20 selected predictor of the dataset and split the data with train_test_split with test_size of 0.3.

	LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1, penalty='l2', random_state=None, solver='liblinear', tol=0.0001, verbose=0, warm_start=False)	KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=5, p=2, weights='uniform')	DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best')	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf', max_iter=-1, probability=True, random_state=None, shrinking=True, tol=0.001, verbose=False)
0	0.613811	0.889441	0.805546	0.894678

Table.2 Test score on 20 informative features of Madelon dataset after dimensionality reduction using 4 different classifiers.

In order the understand the distribution of the 20 selected features, I did statistical analysis on the skew of data. It is clear from the histograms that the each feature has bell curve distribution without any skew centered around the mean. So it is not necessary to deskew the data.

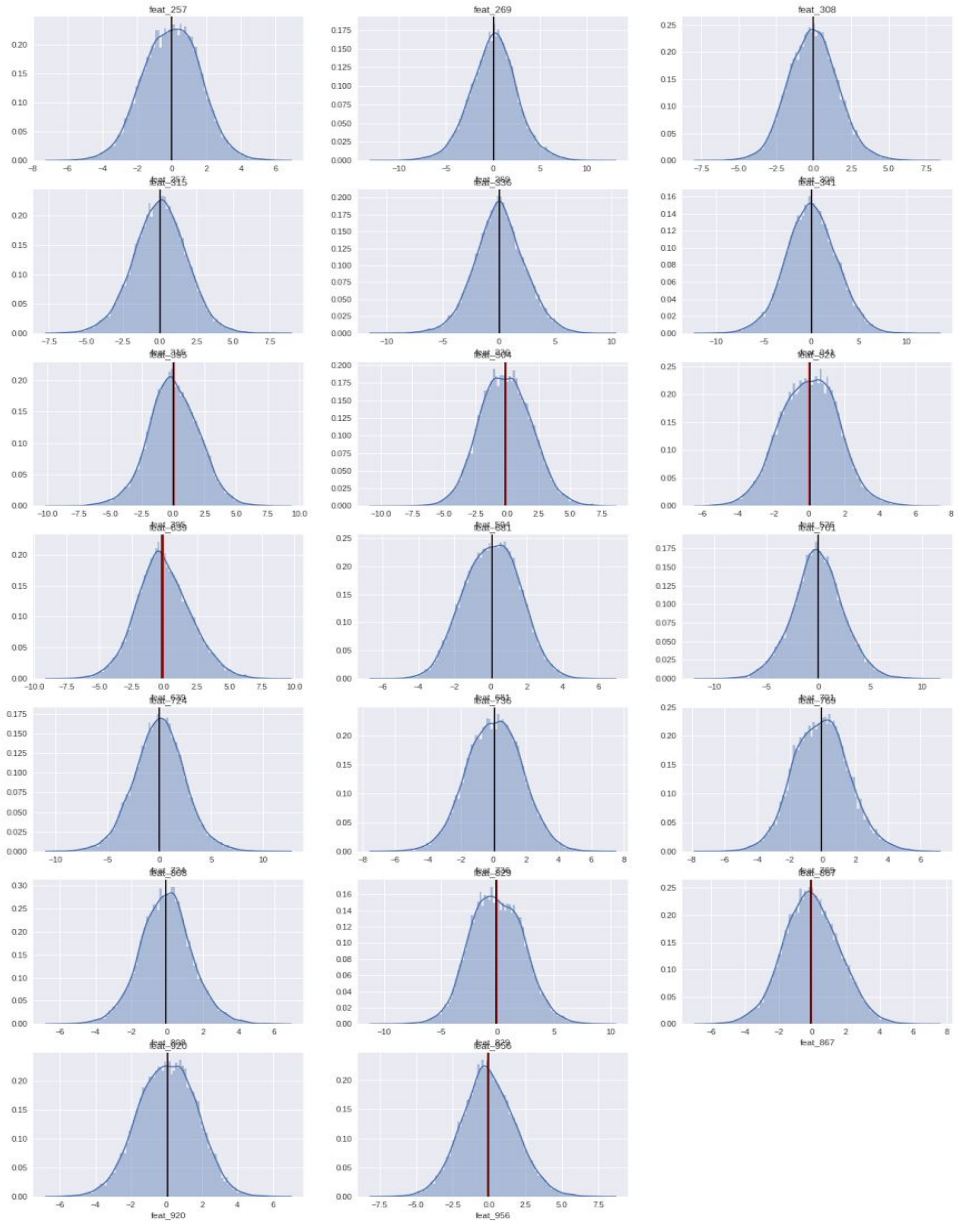

Fig. 4 Distribution plot of the 20 features selected after dimensionality reduction. Vertical red line represents the mean of the feature and vertical blue line represents the median of the feature. All the features have bell shaped curve and do not show any skew in data.

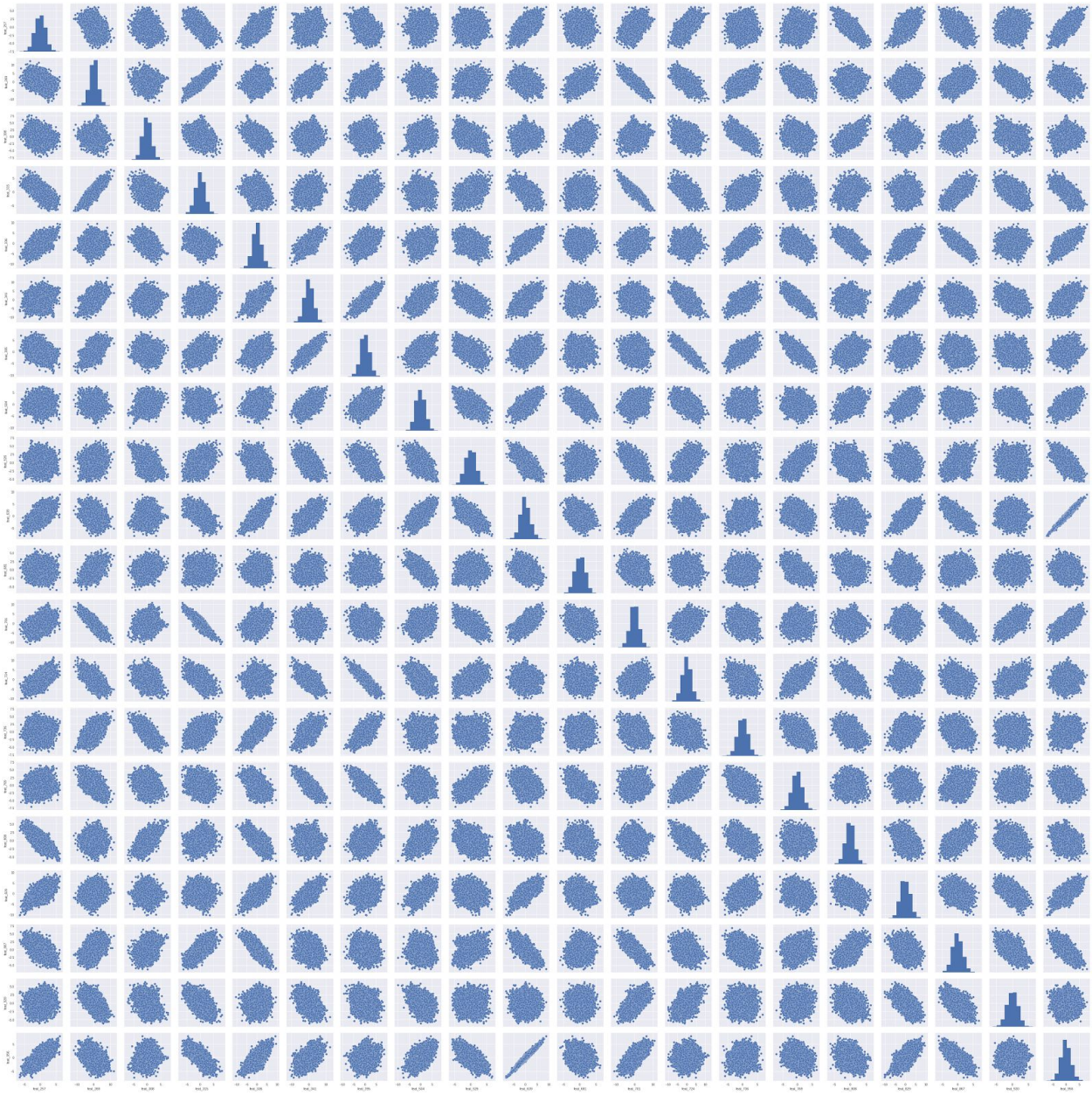

Fig .5 sns pairplot of for the 20 features selected from Madelon dataset after dimensionality reduction

GridSearch classifier for tuning hyperparameters

PCA is a technique to transform data into linearly unrelated components and is useful in reducing the dataset to meaningful components. I am using PCA inside pipeline is order to use the most important features for the classifier.

I created separate pipelines with StandardScaler, PCA and 4 estimators to tune parameters and compared the PR-AUC scores.

1) LogisticRegression: I tuned penalty (l1,l2) and C with range of [0.001, 0.01, 0.1, 1, 10, 100] and trained the data with best parameter (C =100, penalty = l1)

2) KNeighborsClassifier: I tuned n_neighbors in the range of [5,7,9,11,13,15,17,19,21,23,25] and trained the data with best parameter (n_neighbors = 15)

3) DecisionTreeClassifer: I tuned max_depth parameter in the range of [5, 10, 15, 20, 25, 30, 35, 40] and trained the data with best parameter (max_depth = 25)

4) Support Vector Classifier: I tuned C (regularization strength) range(1,100,10) and trained the data with best parameters (C = 11)

	Model_GS	best_estimator	best_param	test_score	train_score
0	LogisticRegression	Pipeline(memory=None,\n steps=[('lr', Logi...	{'lr__C': 100, 'lr__penalty': 'l1'}	0.638475	0.603061
1	KNeighborsClassifier	Pipeline(memory=None,\n steps=[('scaler', ...	{'kn__n_neighbors': 15, 'pca__n_components': 5}	0.911670	0.933424
2	DecisionTreeClassifier	Pipeline(memory=None,\n steps=[('scaler', ...	{'dt__max_depth': 25, 'pca__n_components': 5}	0.813026	1.000000
3	SVC	Pipeline(memory=None,\n steps=[('scaler', ...	{'pca__n_components': 5, 'svc__C': 11}	0.909327	0.947190

Table.3 Score of Madelon dataset after GridSearch using 4 different classifier. The table includes train score, test score, best parameters and best estimators.

Results:

A) 20 Features selected by the 4 statistical methods
[257, 269, 308, 315, 336, 341, 395, 504, 526, 639, 681, 701, 724, 736, 769, 808, 829, 867, 920, 956]

Iterative method of comparing the R2 score had clearly selected all 20 top features as rest of the 980 features R2 scores were really low. Covariance matrix selected 18 of the features with high covariance. SelectKBest with k=100 performed moderately as it selected only 12-13 features whose F-scores were comparable. RandomForestClassifier selected 16-17 of the features whose importance were comparable.

B) After selecting the 20 features from the dataset, I performed GridSearch with separate pipeline made up of StandardScaler, PCA and classifier.

Out of the 4 classifiers (LogisticRegressor, KNeighborsClassifier, DecisionTreeRegressor and SVC), KNeighborsClassifier and SVC performed the best and scored about 90% in PR_AUC.

Conclusion:

I used 4 different dimensionality reduction techniques to reduce the features from 1000 to 20, as Madelon dataset is huge. The 4 different techniques gave 85% similar results, hence I believe I was successful in getting 20 most important features from Madelon dataset.
I used 10% of the data to perform GridSearch to tune the hyperparameter of 4 classifier (LogistcRegressor, KNeighborsClassifier, DecisionTreeRegressor and SVC) in order to get the highest PR_AUC score on test data set.
SVC (C=11) and KNeighborsClassifier(n_neighbor=15) were the best classifier with the PR_AUC score of 90%.