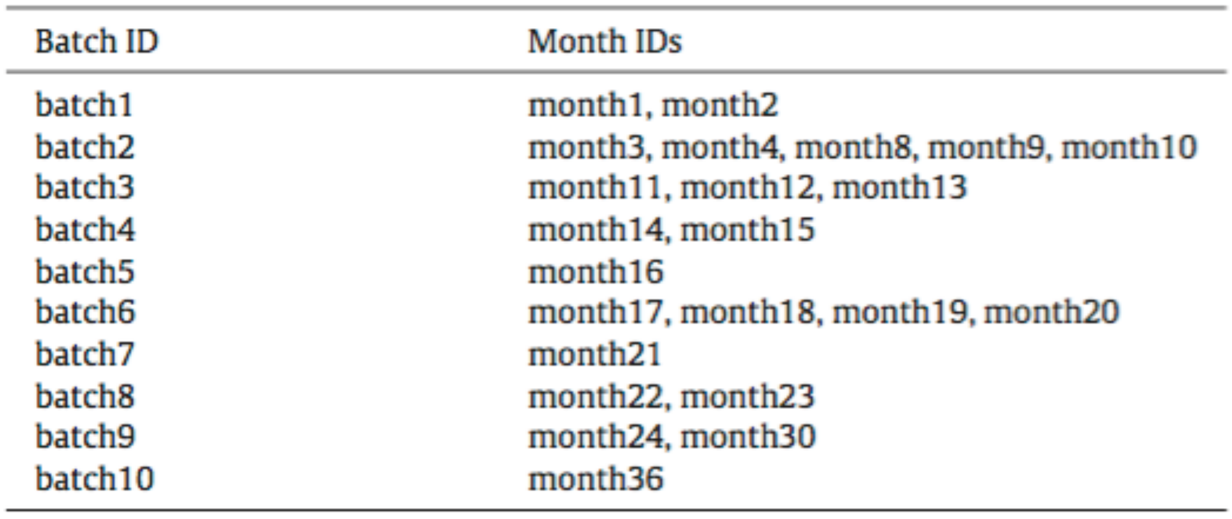
GAS SENSOR ARRAY DRIFT

By Mahrang Saeed

Partner: Devang Kiritbhai Savaliya

Data Set Description:

There are 13,910 measurements taken from 16 chemical sensors that detect 8 features of the following 6 gases: ethanol, ethylene, ammonia, acetaldehyde, acetone, and toluene [1]. The input variables are all the feature values detected by all the sensors, so there are 8 x 16 = 128 input variables. The values of the input variables are the feature values for each measurement recording and they are of type float64. The output variables are the gases, which have been numbered 1-6 in the order listed above, and they are of type int64. There are no missing data. The dataset was gathered over the course of 36 months and is organized into 10 batches in the manner shown in Table 1.



Batch ID	Month IDs
batch1	month1, month2
batch2	month3, month4, month8, month9, month10
batch3	month11, month12, month13
batch4	month14, month15
batch5	month16
batch6	month17, month18, month19, month20
batch7	month21
batch8	month22, month23
batch9	month24, month30
batch10	month36

Table 1: Each row shows the months that were combined to form each batch [2].

Data Set Visualization:

The 128 input variables were reduced to 2 principle components by using Principle Component Analysis in order to visualize the dataset. Batches were added one by one to batch 1 in a sequential order to show the effect of gas sensor drift on the gases over the span of 36 months.

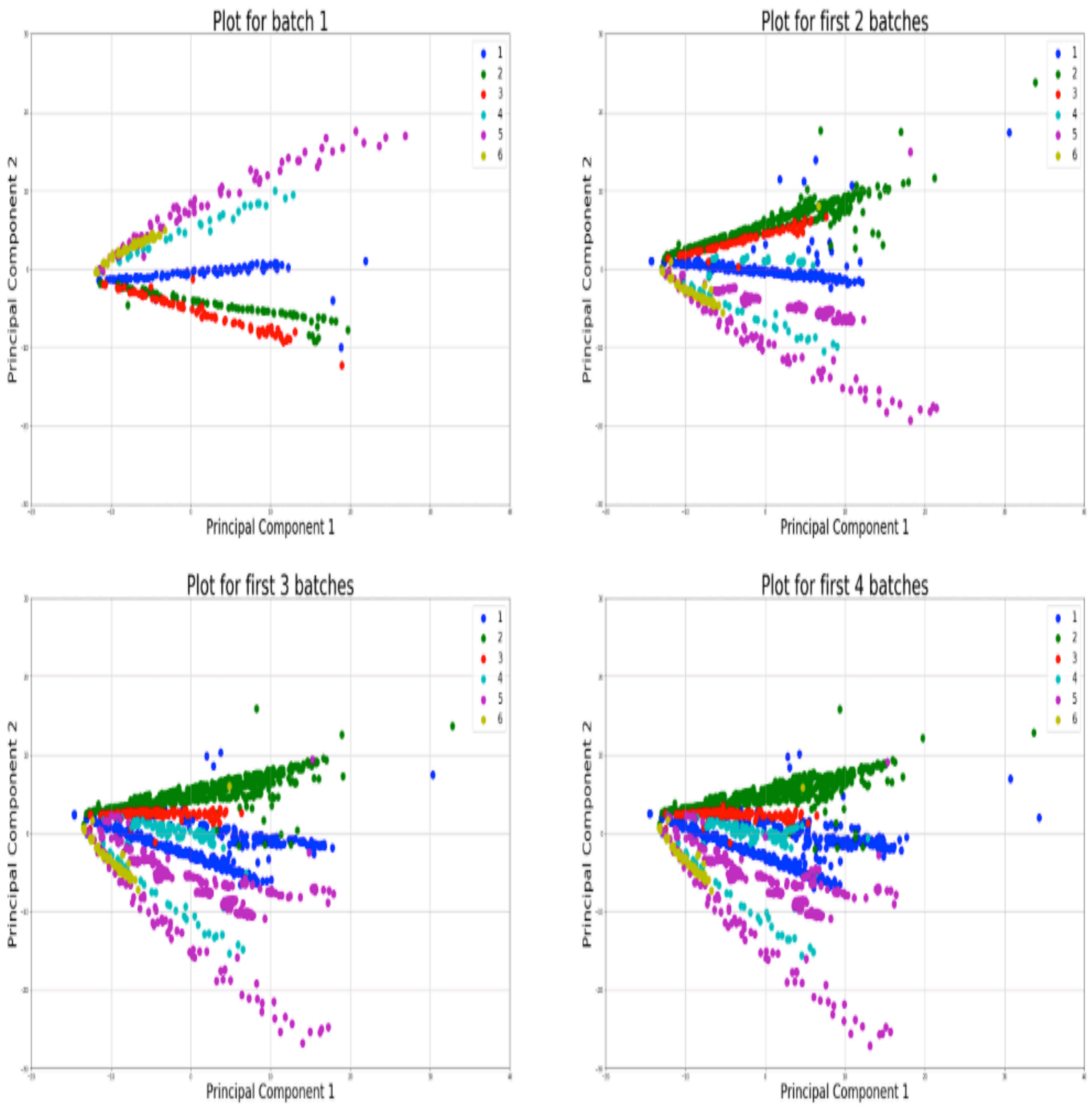


Fig. 1: Effect of gas sensor drift during the first 15 months.

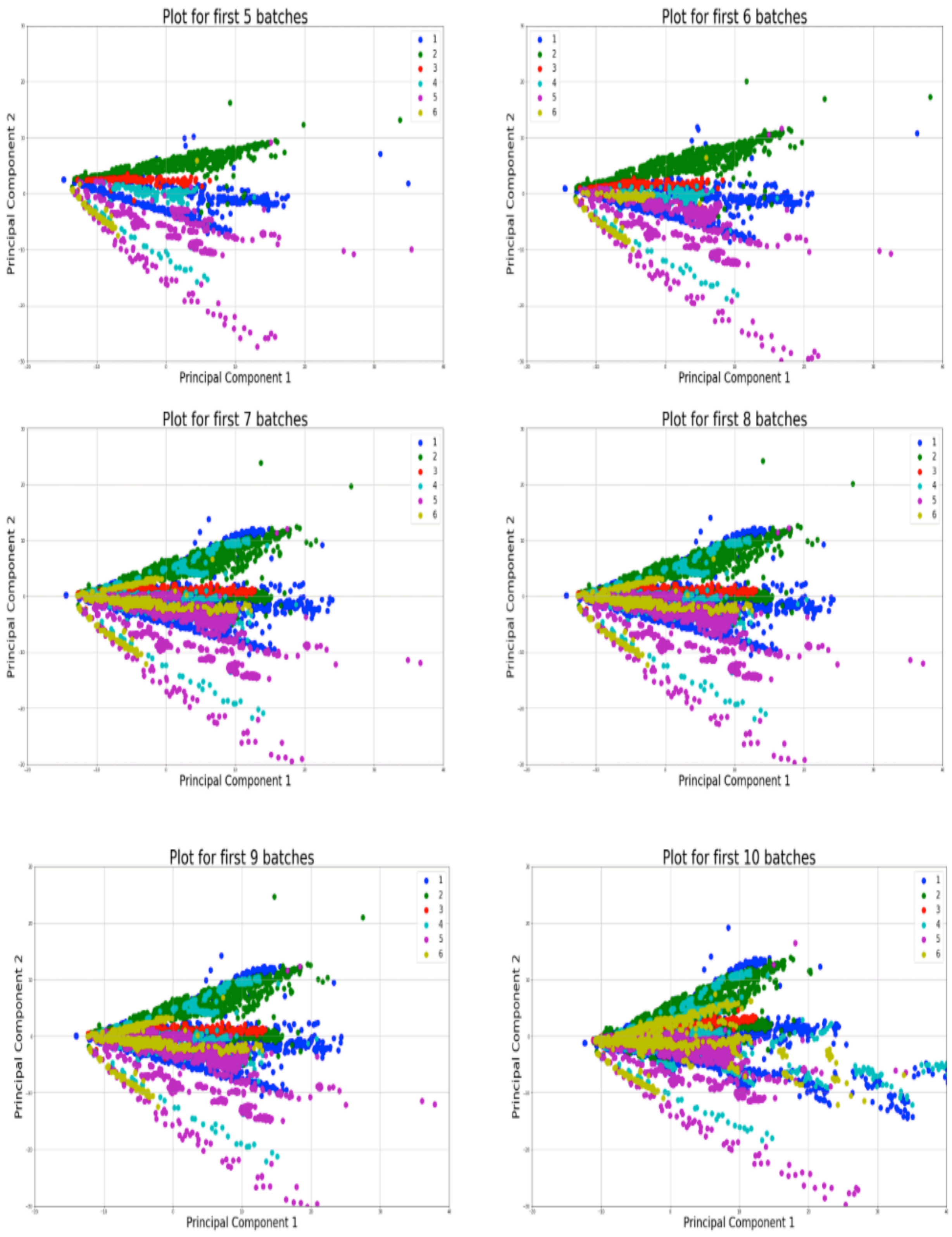


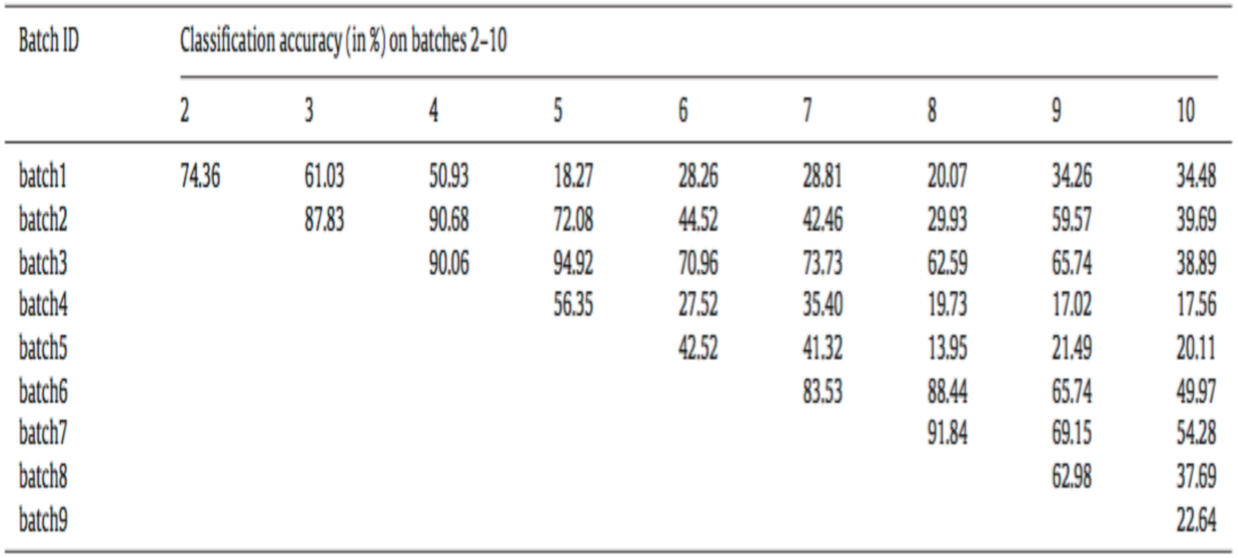
Fig. 2: Effect of gas sensor drift during the entire 36 months.

Data Set Cleaning:

The batches in the dataset were Dat file formatted. We converted the Dat files to Excel files so that we could use them in Jupyter Notebook. There are no outliers in the dataset. Every other column in the dataset was dropped starting from the second column because they were the index numbers of the input variables, with values ranging from 1-128. These index numbers were not needed.

Related Work:

Vergara et. al. used this same dataset to address the problem of gas sensor drift [2]. They used an ensemble of classifiers trained at different points of time and obtained better results than the baseline competing methods. They performed several experiments using the dataset. Their method was to train a multi-class Support Vector Machine, using the one-vs-one strategy, using batches 1-9 and testing on successive batches. This method uses supervised learning algorithm. Their results are shown in Table 2.



Batch ID	Classification accuracy (in %) on batches 2–10								
	2	3	4	5	6	7	8	9	10
batch1	74.36	61.03	50.93	18.27	28.26	28.81	20.07	34.26	34.48
batch2		87.83	90.68	72.08	44.52	42.46	29.93	59.57	39.69
batch3			90.06	94.92	70.96	73.73	62.59	65.74	38.89
batch4				56.35	27.52	35.40	19.73	17.02	17.56
batch5					42.52	41.32	13.95	21.49	20.11
batch6						83.53	88.44	65.74	49.97
batch7							91.84	69.15	54.28
batch8								62.98	37.69
batch9									22.64

Table 2: Performance of classifiers trained on batches 1-9 and tested on successive batches [2].

Feature Extraction:

There is no time info provided in the dataset. There are no dates given. Batches include data from multiple months. Therefore, feature extraction from time series data could not be performed on this dataset.

Model Development:

Three different classification models were used: Logistic Regression, KNN, and Random Forest. The dataset was split into a training set and a test set by using batches 1-9 for training and batch 10 for testing. Then all 128 input variables in both training and test sets were standardized using StandardScaler(). For each classification model, GridsearchCV and cross validation methods were applied to find the best hyperparameters using the test accuracy results. These models were then fine-tuned to find the model with the best performance.

Fine-Tuning Models & Feature Set:

For Logistic Regression, GridsearchCV and 7 fold cross validation methods were applied to find the best hyperparameters. Then PCA was applied to the model with the best hyperparameters to find the best number of principle components, which turned out to be 64 (see Fig. 3). The final Logistic Regression model used was the one that had the best hyperparameters and 64 principle components.

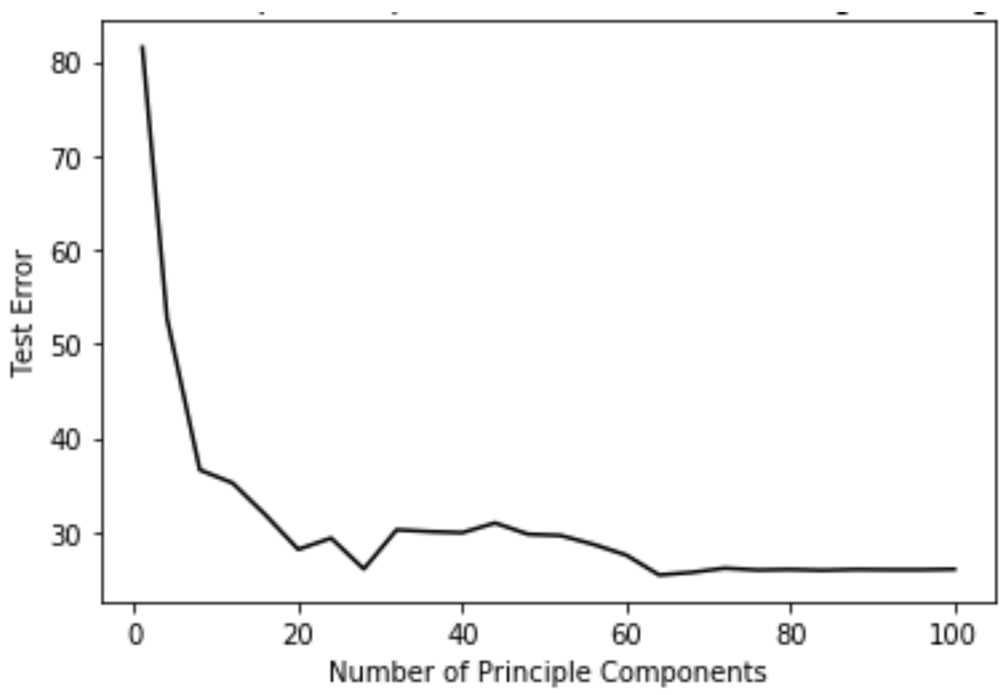


Fig. 3: Number of Principle Components vs Test Error for Logistic Regression.

For KNN, GridsearchCV and 3 fold cross validation methods were applied to find the best hyperparameters. The model with the best hyperparameters was trained with different number of neighbors to find the best number of neighbors, which turned out to be 30 neighbors. Then the model with the best hyperparameters was trained using 30 neighbors. Then PCA was applied to this model to find the best number of principle components, which turned out to be 44 (see Fig. 4). The result is a KNN model with the best hyperparameters, 30 neighbors, and 44 principle components.

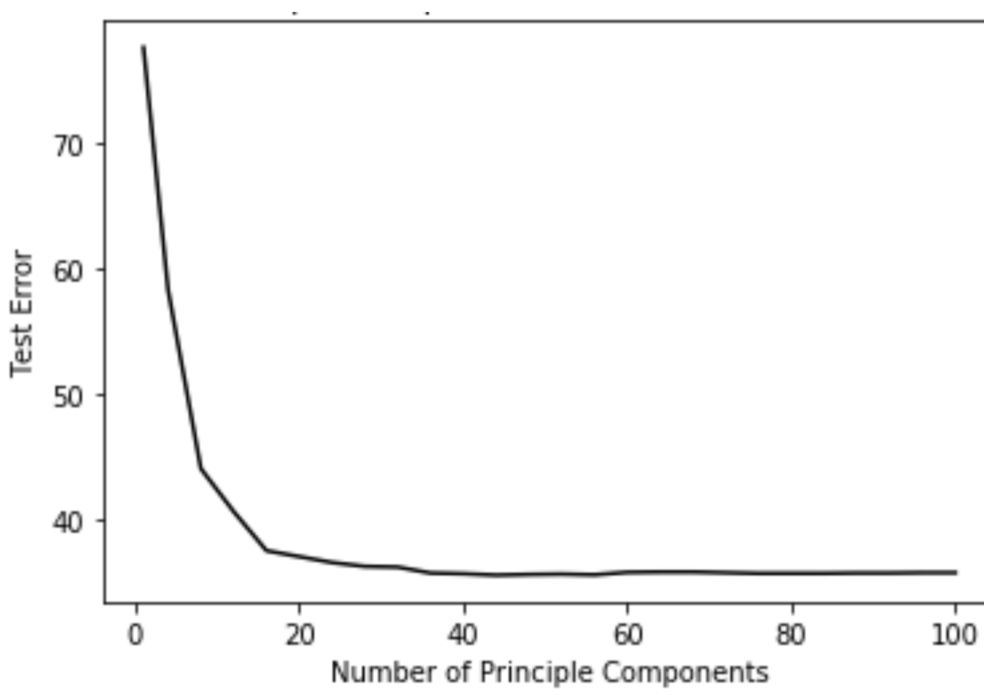


Fig. 4: Number of Principle Components vs Test Error for KNN.

For Random Forest, GridsearchCV and 5 fold cross validation methods were applied to find the best hyperparameters. Different number of trees were tried to find best number of trees, which turned out to be 65 trees. Then PCA was applied to the model with the best hyperparameters and 65 trees to find the best number of principle components, which turned out to be 60 (see Fig. 5). Result is a model with the best number of hyperparameters, 65 trees, using 60 principle components.

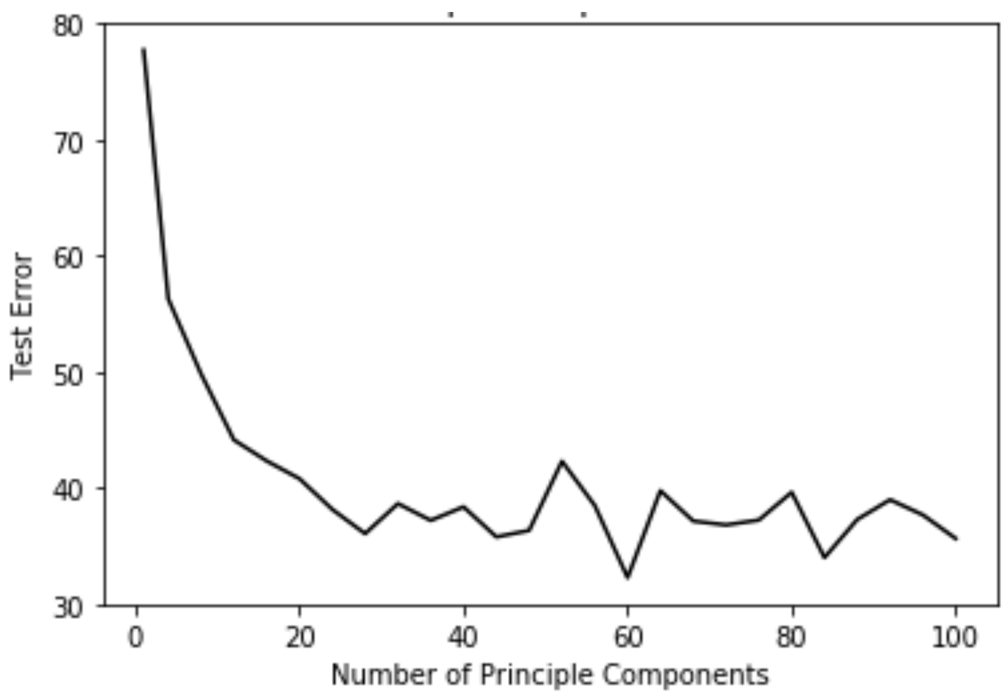


Fig. 5: Number of Principle Components vs Test Error for Random Forest.

Performance:

When all 3 classification models are trained on batches 1-9 and tested on batch 10, the following results are obtained from the fine-tuned models:

	Logistic Regression:	KNN:	Random Forest:
Precision:	78.65%	66.61%	65.52%
Recall:	74.61%	64.42%	62.47%
Test Accuracy:	74.61%	64.4167%	62.47%

Table 3: Performance metrics of the fine-tuned 3 models when trained on batches 1-9.

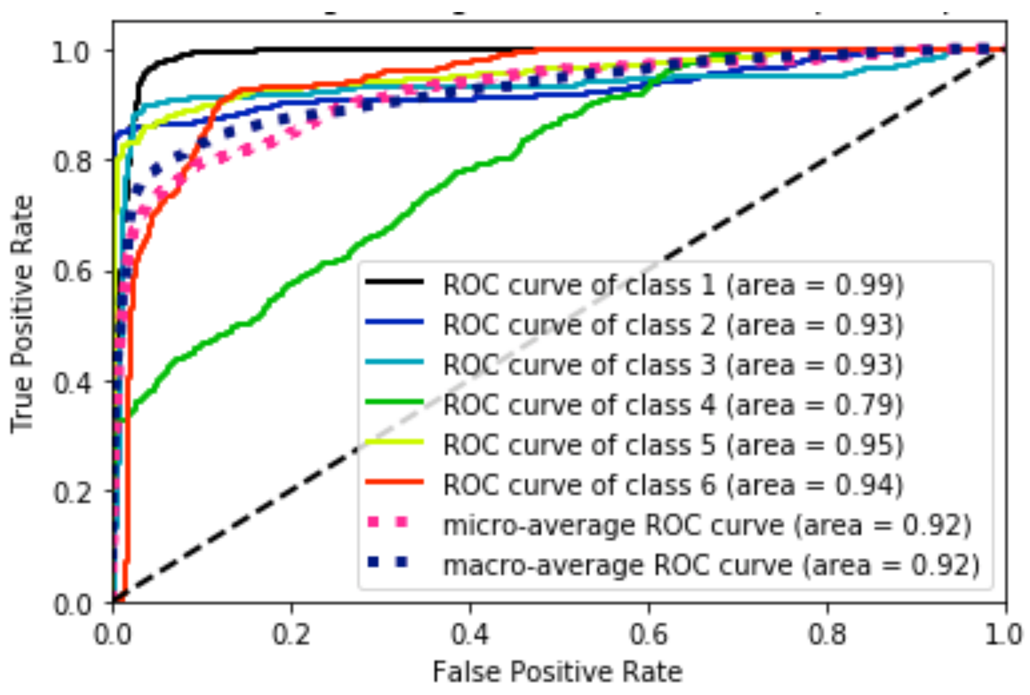


Fig. 6: ROC Curve of the gases for Logistic Regression with 64 principle components.

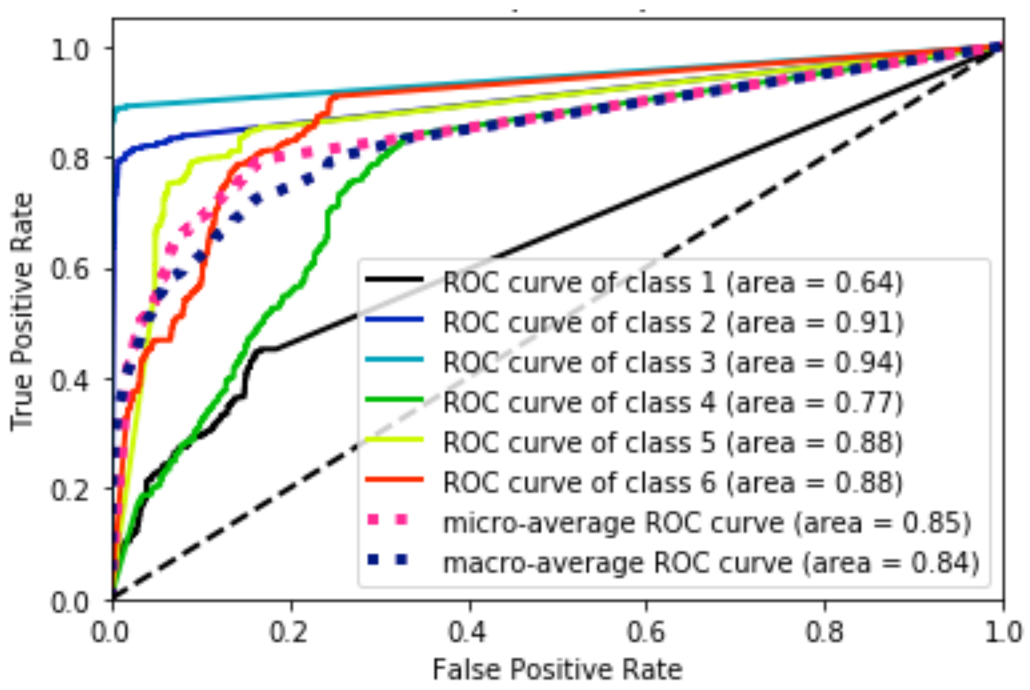


Fig. 7: ROC Curve of the gases for KNN with 30 neighbors and 44 principle components.

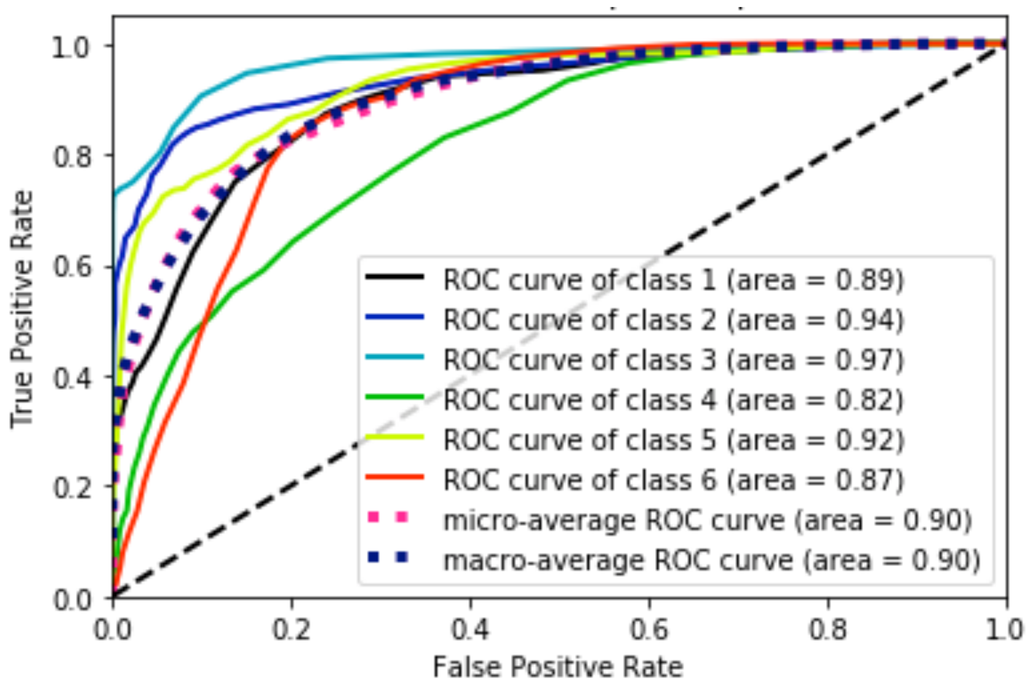


Fig. 8: ROC Curve of the gases for Random Forest with 65 trees and 60 principle components.

<u>Sensor Drift Problem:</u>

As shown by Figures 1-2, as months go by, the gases get mixed over time. This shows that drift exists and classifying the gases at a later time can become harder. In order to be able to classify gases accurately by the end of the 36 months, we need to know which model will have the greatest test accuracy: one that was trained with data from earlier months (batches 1-9), or one that was trained with data from later months (such as batches 6-9). To see the effect of sensor drift on test accuracy, all 3 classification models were trained and fine-tuned again on batches 4-9 and tested on batch 10. The results are shown in Table 4.

	Logistic Regression:	KNN:	Random Forest:
Precision:	74.36%	62.17%	73.83%
Recall:	72.67%	59.08%	65.78%
Test Accuracy:	72.67%	59.08%	65.78%

Table 4: Performance metrics of the fine-tuned 3 models when trained on batches 4-9.

All 3 classification models were also trained and fine-tuned on batches 6-9 and tested on batch 10. The results are shown in Table 5.

	Logistic Regression:	KNN:	Random Forest:
Precision:	72.66%	64.32%	74.9%
Recall:	68.22%	60.97%	71.56%
Test Accuracy:	68.22%	60.97%	72.56%

Table 5: Performance metrics of the fine-tuned 3 models when trained on batches 6-9.

These results show that we get less accuracy when we train Logistic Regression and KNN models on data from later months, which indicate that there is sensor drift. The later batches contain data that is closer to the sensor conditions at the end of the 36 months, which is the test batch 10. Random Forest actually has better performance when trained with batches 6-9 than when trained on batches 1-9.

To get better performance, one could try what Vergara et. al. did in their experiment and use more complex classification models such as Support Vector Machines or the ensemble method.

Conclusion:

Higher accuracy is obtained when training with more data, as in using batches 1-9. Overall, Logistic Regression using batches 1-9 and 64 principle components had the best test accuracy (74.61%) and precision (78.65%) of all the experiments we tried with 3 different classification models and using different number of batches for training. Fig. 6 shows that Logistic Regression is better at classifying most of the gases than the other 2 models. Therefore, Logistic Regression is the best model for classification of gases.

References:

[1] Gas Sensor Array Drift Dataset, UCI Machine Learning Repository, http://archive.ics.uci.edu/ml/datasets/Gas+Sensor+Array+Drift+Dataset

[2] A. Vergara, S. Vembu, T. Ayhan, M.A. Ryan, M.L. Homer, R. Huerta, Chemical gas sensor drift compensation using classifier ensembles, Sensors and Actuators B: Chemical 166-167 (2012) 320-329