Machine Learning Classification of Sarcoma Types using Gene Expression Data -

Project Documentation

### Author: Jeffrey DeLong

## Summary

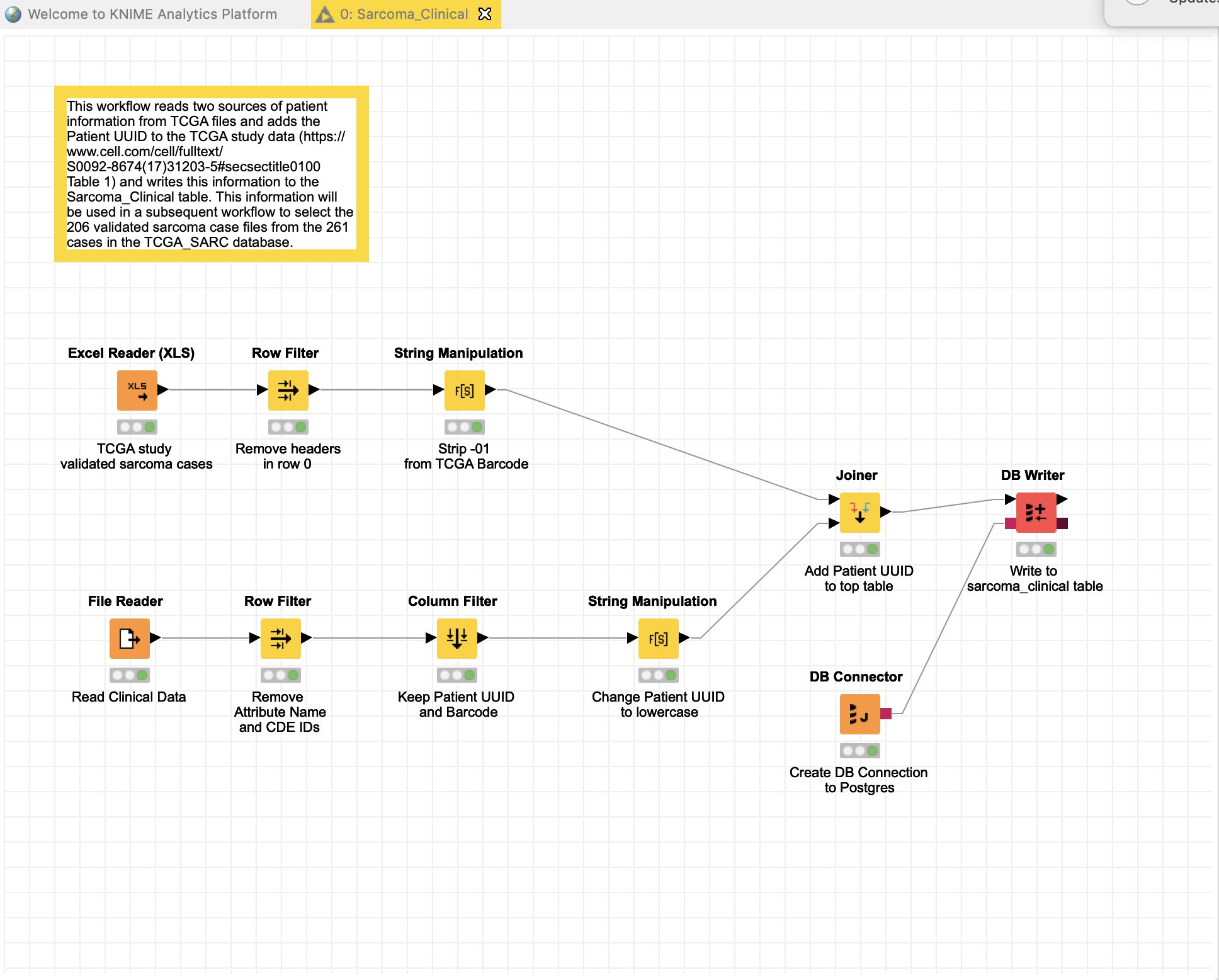
For this study we conducted a series of machine learning analyses of Sarcoma gene expression data using both Knime Analytics Platform Python scikit-learn with Keras with PlaidML. Before conducting the analysis we set up and pre-processed the data in Knime.

## Set up

## Store Clinical Data

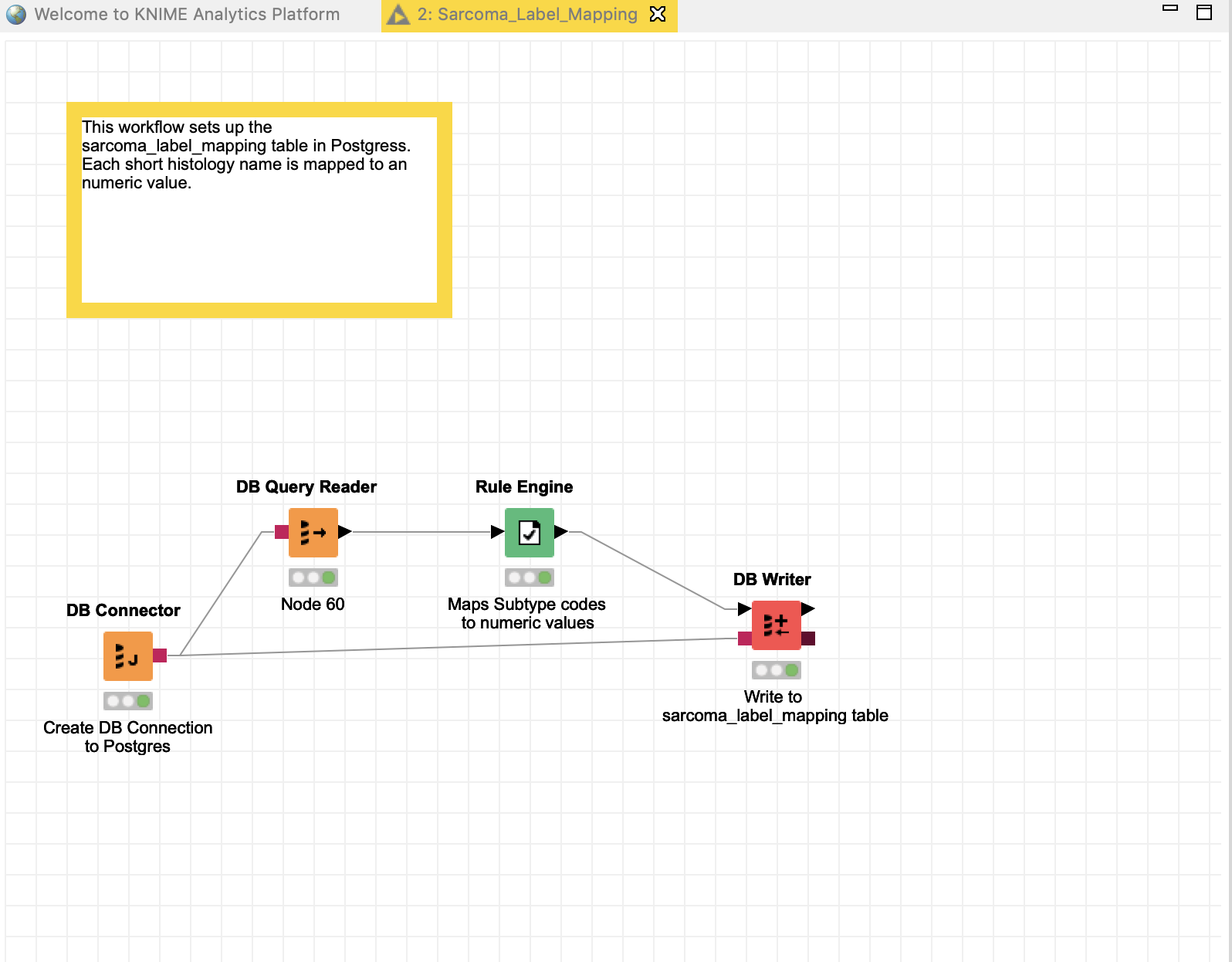
The primary source of data for this analysis is TCGA-SARC project cases found at <https://portal.gdc.cancer.gov/>. In particular the 206 cases analyzed in [4] and [5] are utilized here. The associated patient data was downloaded from

<https://www.cell.com/ell/fulltext/S0092-8674(17)31203-5#secsectitle0100> Table 1. This data is used to identify the 206 cases (from the total 261 cases available in TCGA-SARC). We also downloaded nationwidechildrens.org\_clinical\_patient\_sarc.txt from TCGA-SARC. This file also contains patient information and in particular the Patient UUID which is used to identify the specific gene sequencing data files. The pertinent information from both files was extracted and persisted into a Postgres table, sarcoma\_clinical, using the Sarcoma\_Clinical workflow.

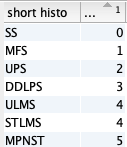


## Create mapping of Sarcoma Type histology designations

Most machine learning algorithms require the class to be expressed as a numeric value (or in some cases, a vector of ones or zeros), so we created the Sarcoma\_Label\_Mapping workflow which maps the Sarcoma Type short histology to a numeric values and persisted this relationship to the sarcoma\_label\_mapping table in Postgres.



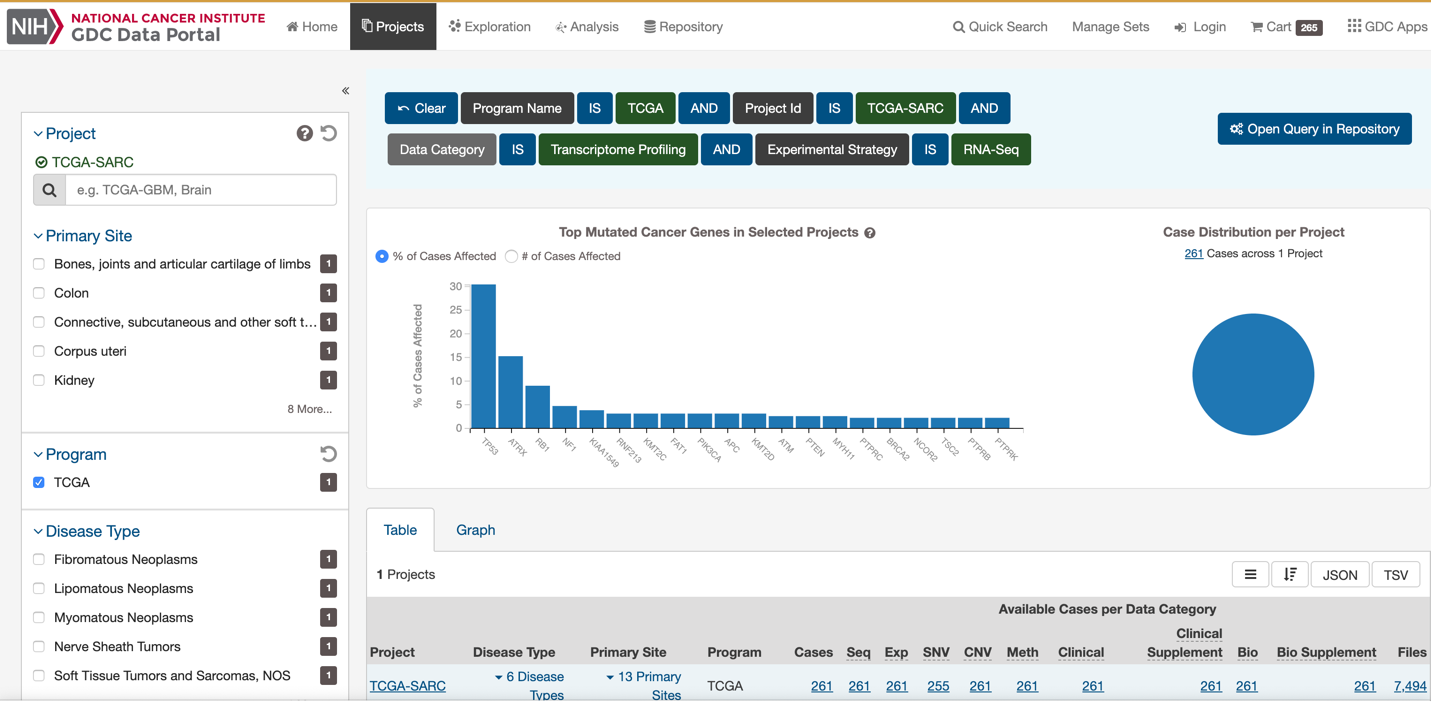
This results in the following mapping:



ULMS and STLMS are both subtypes of LMS, and are therefore labeled the same.

## Download FPKM and FPKM-UQ files

We used the GDC Client command line tool to download the FPKM files from GDC [https://portal.gdc.cancer.gov](https://portal.gdc.cancer.gov/). First select TGCA-SARC as the Project. Then TCGA as the Program, then Data Category Transcriptome Profiling, then Experimental Strategy of RNA Sequencing.

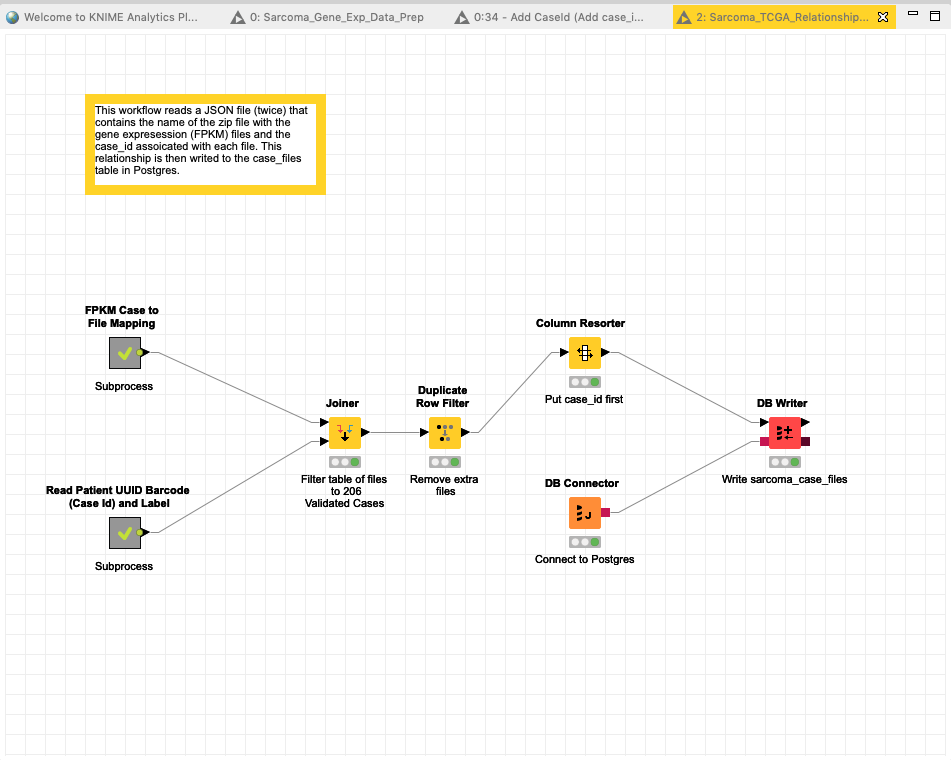


Then select the Open Query in Repository, and select HT-Seq - FPKM). Once selected, first click JSON bottom top right, which will download a JSON file that maps the file\_name to the case\_id (this file is used below). Then select Add All Files to Cart (top left), and download the manifest.

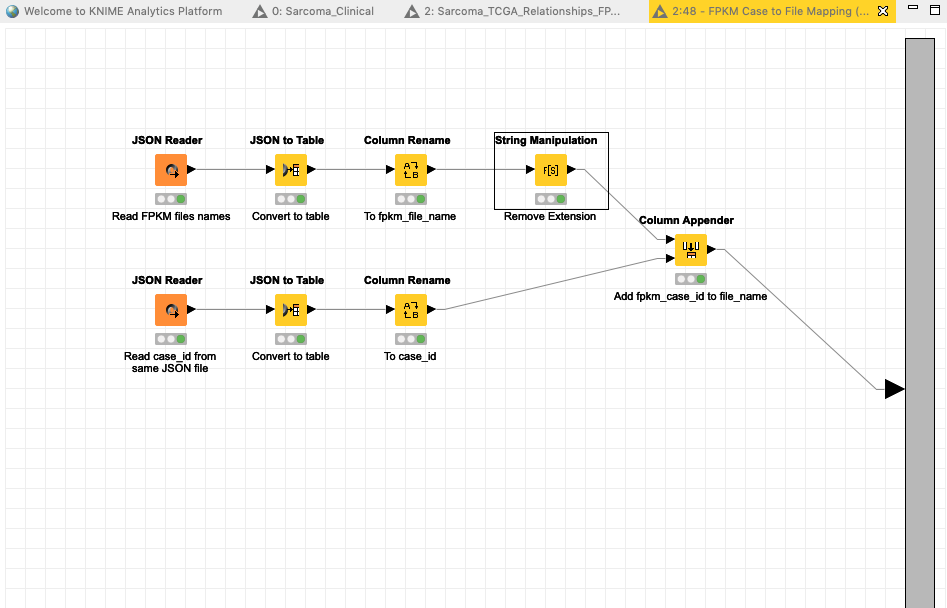
Open a command prompt, cd into the directory to download the files into, and execute PATH\_TO\_GDC\_CLIENT/gdc-client download -m PATH\_TO\_GDC\_MANIFEST. This will download the FPKM files. Unzip the files once they are downloaded.

## Create mapping of FPKM files to Case Id

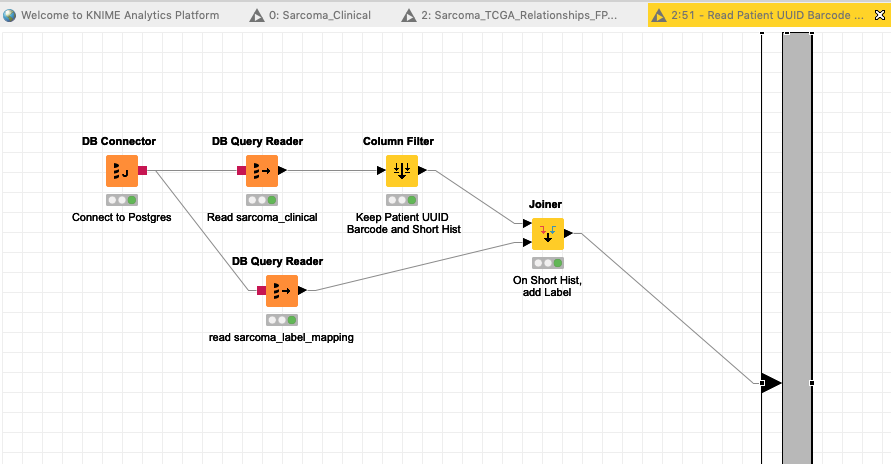
We then used the JSON file that associates each gene expression file with a case Id to build a workflow, Sarcoma\_TCGA\_Relationships\_FPKM, that creates a database table, sarcoma\_case\_files, with the caseid, fpkm file name, and label.



This workflow has two sub-workflows:



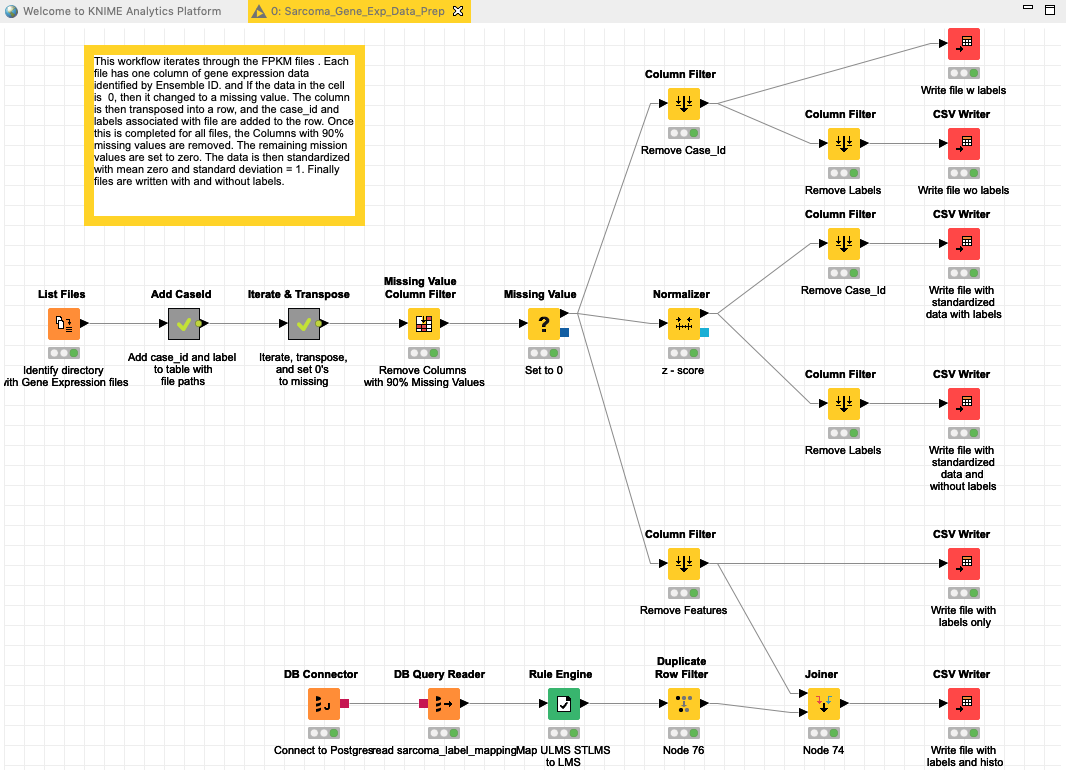
And



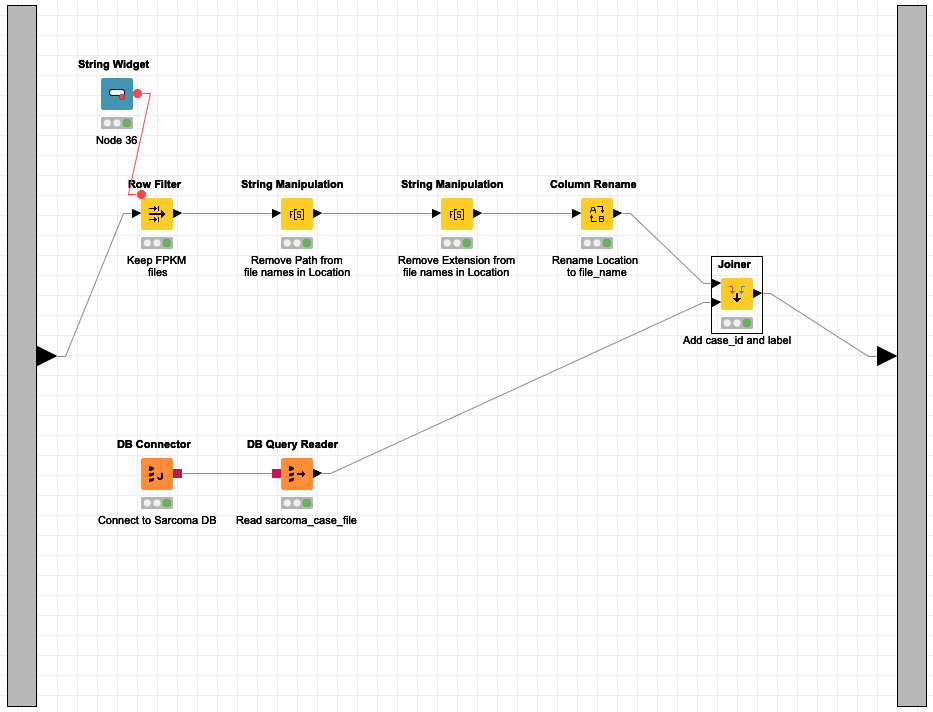
Note for unknown reasons the 259 cases have 265 files associated with them. We remove the duplicate files so that each case has only one FPKM file.

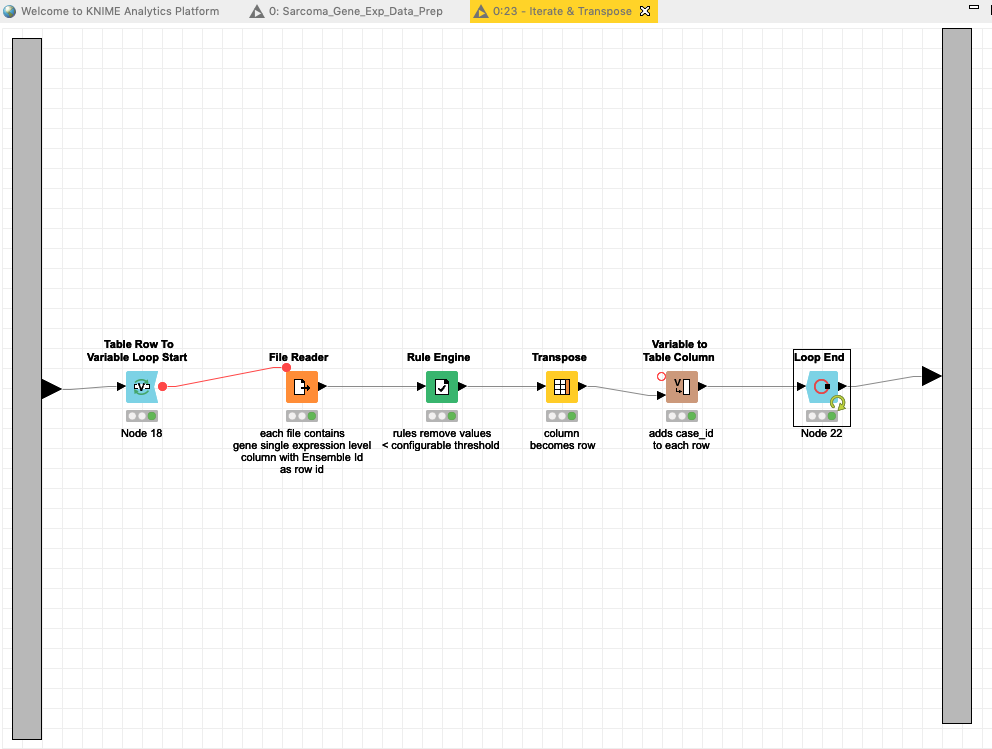
## Data Preprocessing

The Sarcoma\_Gene\_Exp\_Data\_Prep workflow iterates through the FPKM files. Each file has one column of gene expression data identified by Ensemble ID. and If the data in the cell is 0, then it changed to a missing value. The column is then transposed into a row, and the case\_id and labels associated with file are added to the row. Once this is completed for all files, the Columns with 90% missing values are removed. This reduces the number of features to 20605. The remaining missing values are set to zero. The data is then standardized with mean zero and standard deviation = 1. Files are then written with and without labels.



This workflow has two sub-workflows:





## Data Analysis and Feature Reduction

We used both non-standardized and standardized data to perform both unsupervised and supervised learning experiments using Python skikit-learn on the University of Utah SOM Server in order to better understand the data, compared to results from previous authors, and identify opportunities to reduce the number of features for subsequent machine learning analysis. In particular we performed Chi Squared analysis, K-means Clustering, Hierarchical Clustering, Principal Component Analysis, and t-Stochastic Neighbor Embedding.

We then executed several machine learning models using Python skikit-learn, including K Nearest Neighbor, Support Vector Machine, Random Forest, and Logistic Regression (with solver SAG) on the various feature reduced subsets identified by these techniques. All data was split 75 / 25 training vs test, and K-Fold cross validation was used for training with four folds.

## Chi Squared

The Jupyter Notebook Sarcoma-FeatureSelection-Chi2.ipynb was used to perform the Chi Squared analysis on sarcoma-gene-exp-FPKM-no-labels.csv and sarcoma-gene-exp-FPKM-labels-only.csvto find the 200 best features.

chi\_best = SelectKBest(chi2, k=200)

sarcoma\_fs = chi\_best.fit\_transform(sarcoma\_df, sarcoma\_labels\_df)

Two csv files were created: sarcoma-gene-exp-FPKM-chi2-gene-symbols-list.csv and sarcoma-gene-exp-FPKM-chi2-features.csv.The first with a list of the gene symbol names (translated from Ensembl Ids using MyGene service) and the second with the data associated with those features.

GridSearchCV was then performed in the Jupyter Notebook Sarcoma-GridSearchCV-Chi2.ipynb with the new dataframe created from sarcoma-gene-exp-FPKM-chi2-features.csv

using the 4 classifiers. This required the features be first z-score standardized. The results are in the following table.

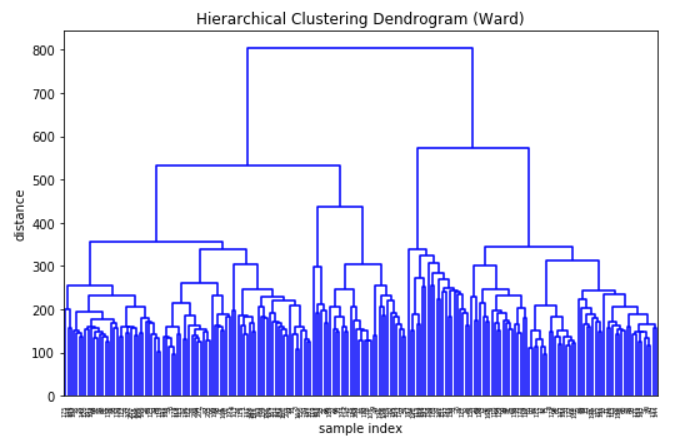
|  |  |  |  |
| --- | --- | --- | --- |
| Model | Parameter Name | Best Parameter Setting | Test Accuracy Score |
|  |  |  |  |
| KNN | # of Neighbors | 5 | .73 |
| SVM | C, gamma | 1, .001 | .79 |
| RF | Max Depth | 4 | .77 |
| LR | C | 2.78 | .79 |

## Clustering

### Hierarchical Clustering with 6 clusters

Hierarchical cluster was performed with the Jupyter Notebook Sarcoma-HC.ipynb using the z score standardized data files sarcoma-gene-exp-FPKM-zscore-no-label.csv

We first drew a dendogram using scipy.cluster.hierarchy.dendrogram



We then performed hierarchical agglomerative clustering with affinity='euclidean' and linkage='war' with the number of clusters set to 6.

A cluster dataframe was created by concatenating the cluster labels for each observation with the short histo, read from sarcoma-gene-exp-FPKM-hc-histo.csv, and then grouping these:

results = cluster\_df.groupby(["cluster", "short histo"]).size()

This gave a count of each type of sarcoma within each cluster, which was written to sarcoma-gene-exp-FPKM-hc-histo.csv for further analysis. The purity of this cluster is .73

### K-Means Clustering with 6 clusters

K-means clustering was performed with the Jupyter Notebook Sarcoma-KMeans.ipynb using the z score standardized data file sarcoma-gene-exp-FPKM-zscore-no-label.csv. The resulting inertia value is 342667

Similar to Hierachical Clustering, a cluster dataframe was created and grouped and written to sarcoma-gene-exp-FPKM-kmeans-histo.csv. Purity was calculated at .62.

### Clustering – 3 clusters

The two clustering algorithms were also run with number of clusters equal to 3. Both approaches identified the following three clusters:

1. MPS, UPS, and DDLPS clustered together
2. LMS (ULMS and STLMS) clustered together
3. Mix of DDLPS, SS and MPNST

## Principal Component Analysis

Principal Component Analysis was performed in the Jupyter Notebook Sarcoma\_PCA.ipynb on the z-score standardized FPKM data read from sarcoma-gene-exp-FPKM-zscore-no-label.csv using Python skikit-learn with the variance set to .95. This resulted in 163 components. I.e., 163 components preserve 95% of the variance in the 20000 plus gene expression levels.

The principal components were combined with the associated labels and used as the features for a series of machine learning algorithms using Grid Search in the Jupyter Notebook Sarcoma-GridSearchCV-PCA.ipynb with the following results:

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Parameter Name | Best Parameter Setting | Test  Accuracy Score |
|  |  |  |  |
| KNN | # of Neighbors | 5 | .60 |
| SVM | C, gamma | 100, 1e-05 | .79 |
| RF | Max Depth | 5 | .60 |
| LR | C | 1.0 | .83 |

### PCA with Hierarchical Cluster added as a Feature

We then added the cluster number from the Hierarchical Cluster (6 cluster centers) to the component feature set in Sarcoma-GridSearchCV-PCA-Cluster.ipynb and ran a grid search with the following results:

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Parameter Name | Best Parameter Setting | Accuracy Score |
|  |  |  |  |
| KNN | # of Neighbors | 5 | .69 |
| SVM | C, gamma | 10, 1e-05 | .79 |
| RF | Max Depth | 9 | .63 |
| LR | C | 1.0 | .81 |

## T-Stochasitic Neighbor Embedding

T-Stochastic Neighbor Embedding was performed in Sarcoma-TSNE.ipynb on the z-score standardized FPKM data with the number of components set to 2, a perplexity of 60 and an angle of 0.5 using the Barnes-Hut algorithm. After the two components were calculated, the label data was added, the data was grouped based on label values (sarcoma types), the mean of each group was calculated, and the means plotted. The resulting plot show UPS, MFS, and DDLPS clustered near one another, and MPNST, LMS, and SS spread apart.

Add plot

The two components were written to sarcoma-gene-exp-FPKM-tsne2.csv.

The notebook Sarcoma-GridSearchCV-TSNE.ipynb reads the sarcoma-gene-exp-FPKM- tsne2.csv, and then executes Grid Search with the four models with the following results:

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Parameter Name | Best Parameter Setting | Accuracy Score |
|  |  |  |  |
| KNN | # of Neighbors | 7 | .71 |
| SVM | C, gamma | 1.0, scale | .61 |
| RF | Max Depth | 5 | .63 |
| LR | C | 1 | .69 |

## Artificial Neural Network

## Baseline

A Multi-layer Perceptron model was created using Python and Keras with PlaidML backend in Sarcoma-MLP-Keras.ipynb. This baseline model was used to classify sarcoma types with no tuning or feature extraction.

Model data is 206 sarcoma cases with 20605 features, which had been z-score normalized. Data was split 60/20/20 between training, validation, and test. Accuracy was used to evaluate the models. All models defined four layers with the hyperparameters as defined below.

### Model and Hyperparameters

|  |  |  |
| --- | --- | --- |
| **Layer** | **# of output units** | **Activation Function** |
| Input Layer | 5000 | ReLU |
| Hidden Layer (Keras / DL4J Dense Layer) | 5000 | ReLU |
| Hidden Layer Keras / DL4J Dense Layer) | 5000 | ReLU |
| Output layer Keras / DL4J Dense Layer) | 6 | ReLU  Loss Function: Multiclass Cross Entropy |

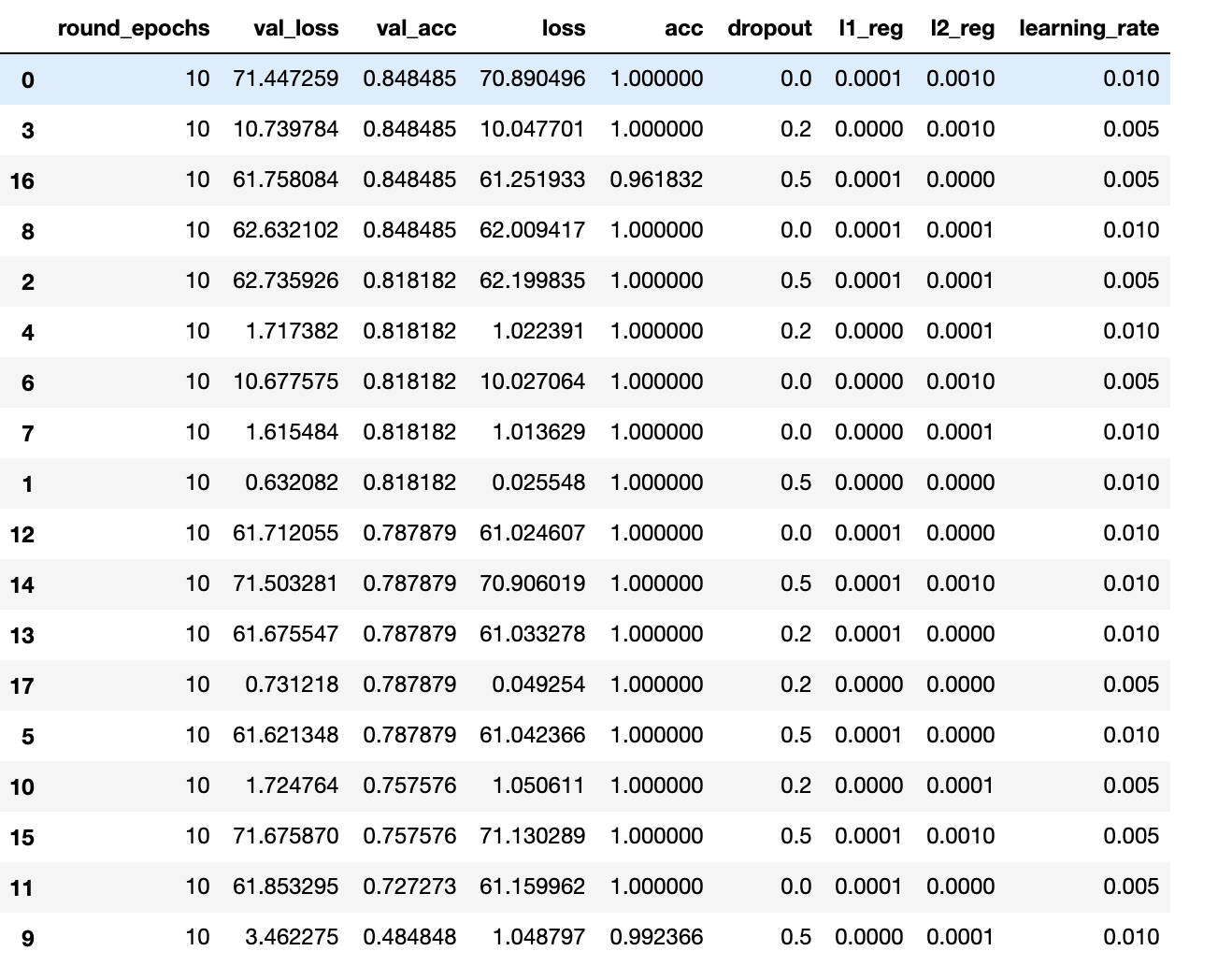
|  |  |
| --- | --- |
| **Hyperparameter** | **Value** |
| Batch Size | 32 |
| Epochs | 10 |
| Global Learning Rate | .01 (Default) |
| Optimizer | Stochastic Gradient Descent |

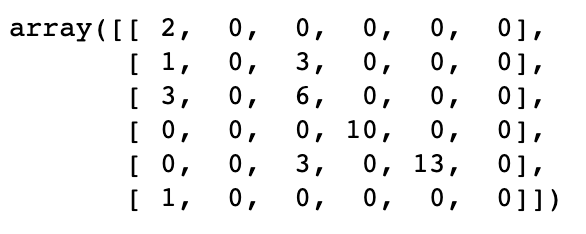
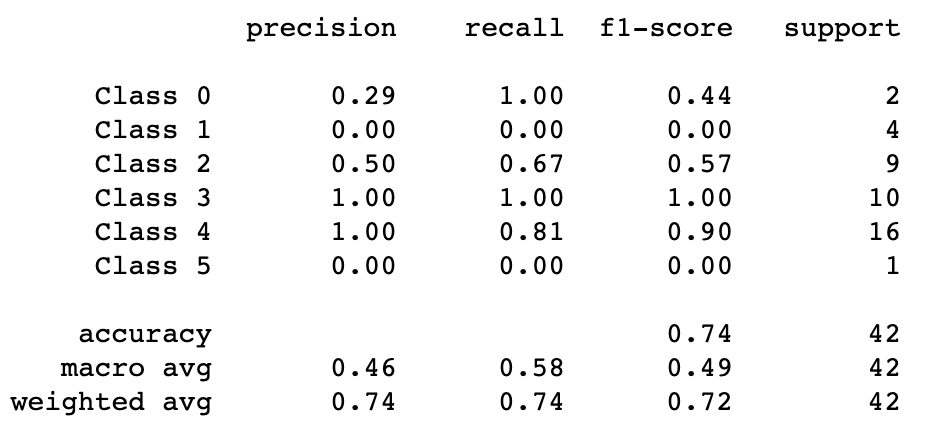
This model has a validation accuracy of 80%

Several variations of this model were examined. These included Sarcoma-MLP-Keras-Reg-L1.ipynb, Sarcoma-MLP-Keras-Reg-L2.ipynb, and Sarcoma-MLP-Keras-Dropout.ipynb.

## Hyperparameter Tuning

Hyperparameter tuning was conducted using Talos in the notebook Sarcoma-MLP-Keras-Talos.ipynb. Four parameters were tuned: Learning Rate, l1 regularization, l2 Regularization, and Dropout.





We took the best model hyperparameters and evaluated it against the test data in Sarcoma-MLP-Keras-Final.ipynb.

All models consistently mis-classified MFS. Most of the MFS sample were misclassified as UPS. This result is explained by the similarity between UPS and MFS, which are some people consider the same disease, the only significant difference being larger amounts of myxoid stroma in MFS tumors [4].

We removed the MFS samples from out data set, re-computed the Principal Components, and re-ran the four machine learning models on the new set of principal components in Sarcoma-GridSearchCV-PCA-No-MFS.ipynb. Logistic Regression was the best model. With SAG solver and C of 1 we achieved a test data accuracy of .88. We then created a BaggingClassifier with the Logistic Regression classifier with n\_estimators=100 and max\_samples=100, and achieved a test data accuracy of 92%,

As a final step we utilized SMOTE with a sampling strategy of ‘not majority’ to provide 48 samples of each label with the Bagging Classifier with Logistic Regression to achieve an accuracy of 97% in Sarcoma-LR-PCA-No-MFS-IMB-Bagging-Final.ipynb

