ANOMALY DETECTION EXERCISES

Thomas G. Dietterich, Chief Scientist, BigML and Professor (Emeritus), Oregon State University
Guillem Vidal, Machine Learning Engineer, BigML

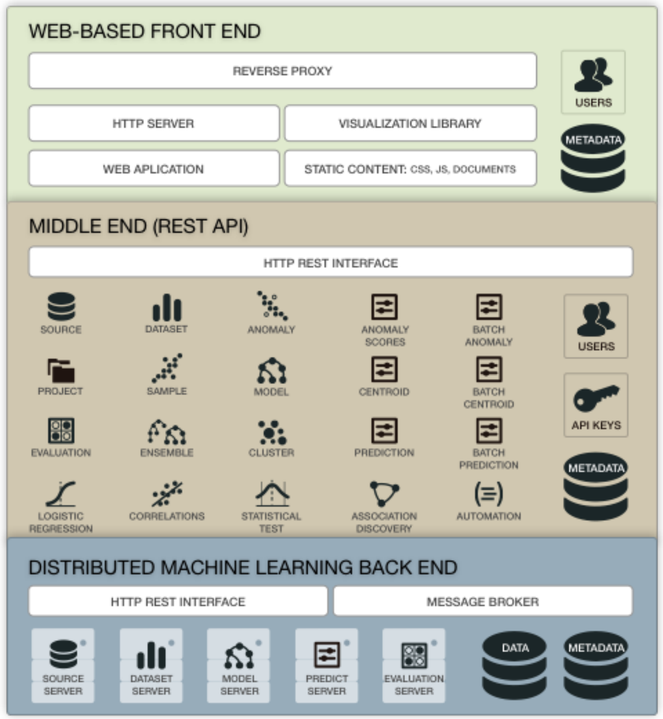
Agenda

1	Exercise 1: Calibration
2	Exercise 2: Novel Category Discovery
3	Exercise 3: Fraud Detection
4	Exercise 4: Data Cleaning

BigML Interface Introduction

SHORT DEMO

- BigML API first architecture
- BigML resources are unmutable
- Remember to pick good names for your resources



Agenda

1	Exercise 1: Calibration
2	Exercise 2: Novel Category Discovery
3	Exercise 3: Fraud Detection
4	Exercise 4: Data Cleaning

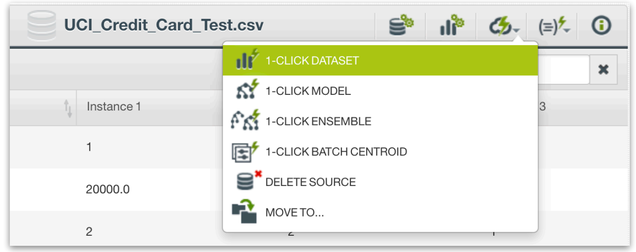
Exercise 1

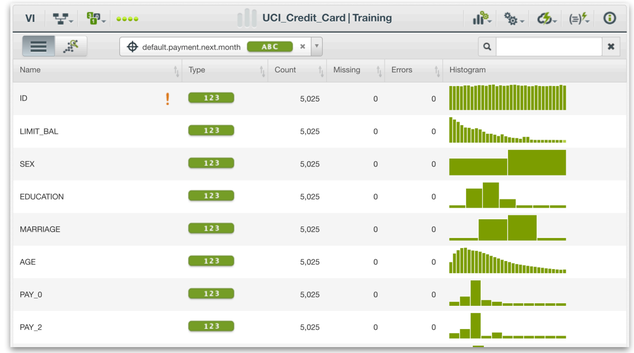
UCI CREDIT CARDS DATASETS

- 3 datasets provided: training, calibration and test
- About 5,000 instances each, with 25 fields. Each instance represents a credit card
- Volume 500 KB
- default.payment.next.month is the objective field and corresponds to a boolean value indicating if the given credit card will default in the next month

INSTRUCTIONS:

- Drag and drop the 3 files in the browser BigML page logged into your account
- A source will be created for each file in the Sources tab. Click on the source to open it
- From each source create a dataset (In the one-click menu select the one-click dataset option, adataset will be created in the datasets tab)





Exercise 1

STEPS

- Split the dataset in 3 parts: train, calibration and test
- Train a boosted decision tree over the training set
- Verify if the resulting probabilities are calibrated
- Apply plaat scaling calibration
- Predict and calibrate over the test dataset
- Verify final results

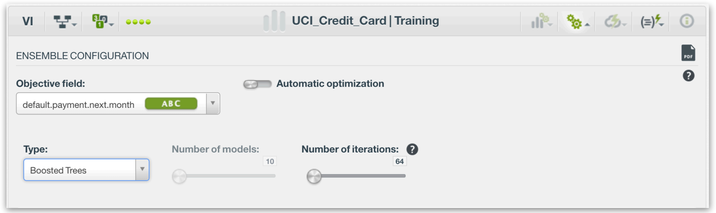
Boosted Decision Tree Classification

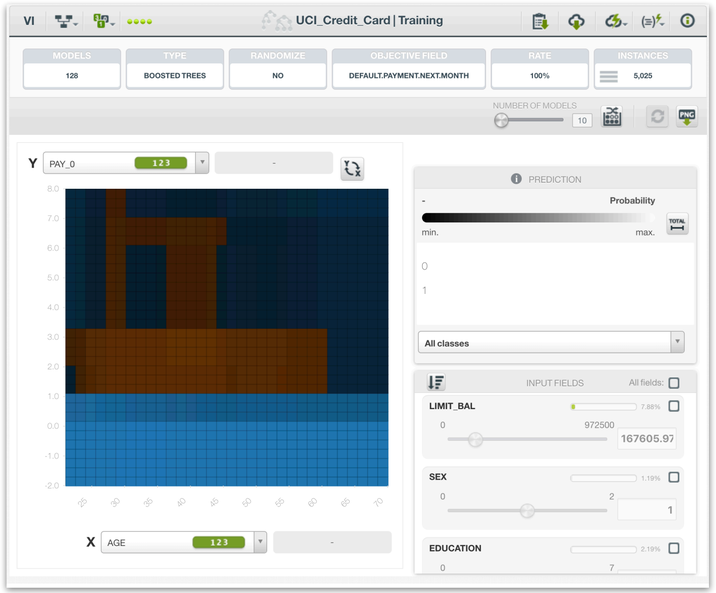
TRAIN A BOOSTED ENSEMBLE TO CLASSIFY THE DEFAULT

- Let’s first train a boosted tree ensemble to classify default payments for each credit card

INSTRUCTIONS:

- Open the training dataset and in the configurations menu (gears icon at the top right) select the “Ensemble” option. In BigML Ensembles are groups of decision trees, they cover random forests and boosted trees. The configuration panel will appear
- In the configuration set the type selector to “Boosted Trees”, set the number of iterations to 120 and click on the create ensemble button. This will train a boosted tree ensemble with up to 120 decision trees sequentially
- The new ensemble will appear including visualizations. Explore the graphics. Boosted trees are represented with a Partial Dependence Plot (PDP) where model results are colored based on 2 variable dimensions and additional feature values can be set on the bottom right too
- The “Ensemble Summary Report” (first icon on the top right menu) displays the fields importance





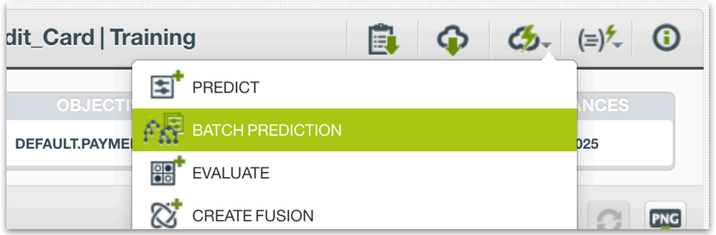
Batch Prediction

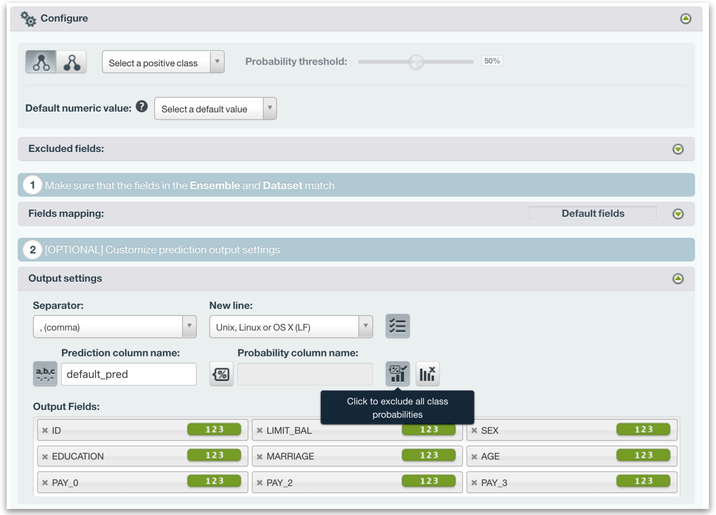
BATCH PREDICTION OVER THE CALIBRATION SET

- We will need to do a batch prediction to obtain prediction probabilities over the calibration dataset
- Resulting probabilities will be used later for calibration

INSTRUCTIONS:

- On the cloud menu select the “Batch Prediction” option (see screenshot on the right). Select the calibration dataset as target
- The output needs to be configured to extract probabilities for both classes. Before clicking on the “Predict” button, in the configuration panel’s output section set the new predicted field name as “class_pred” and click on the button to include all class probabilities (see screenshot 2)
- This will generate a dataset including 3 new fields, a preview is available as well. Name the dataset so it can be easily identified, with a “pred” suffix for example
- Open the resulting dataset in BigML





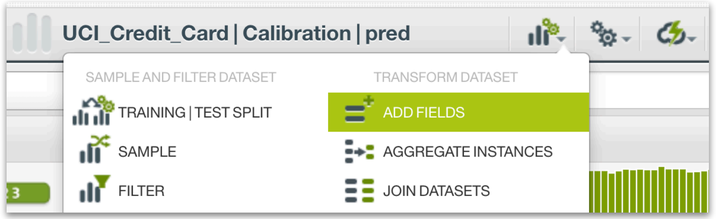
Reliability Diagram

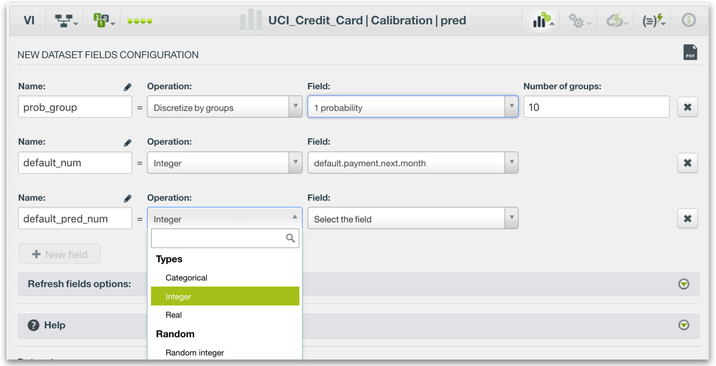
DATA PREPARATIONS: PROBABILITY GROUPS

- To plot a reliability diagram in BigML we will need to proceed with several transformations
- First calibration dataset instances need to be grouped into 10 bins of probability equal ranges

INSTRUCTIONS:

- On the batch prediction resulting dataset, let's add a new field with the probability group information. For this in the transformations menu on the top right select the "Add Fields" option (see screenshot 1)
- In the configuration panel appearing a selector with different pre built operations appear. Select the option “Discretize by groups” for the 1_probability field and define 10 groups. The new field can be named prob_group (see screenshot 2)
- Additionally add another field casting default field type into numeric (we will need to sum later). Add a new field with the option Type (Integer) in this sense (see screenshot 2)





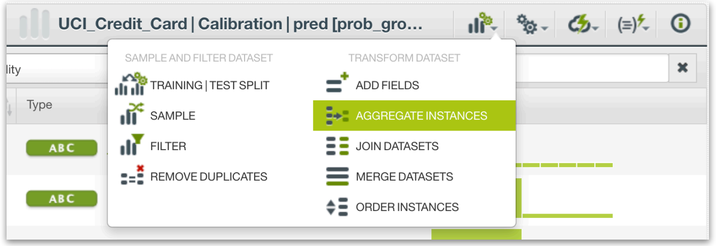
Reliability Diagram

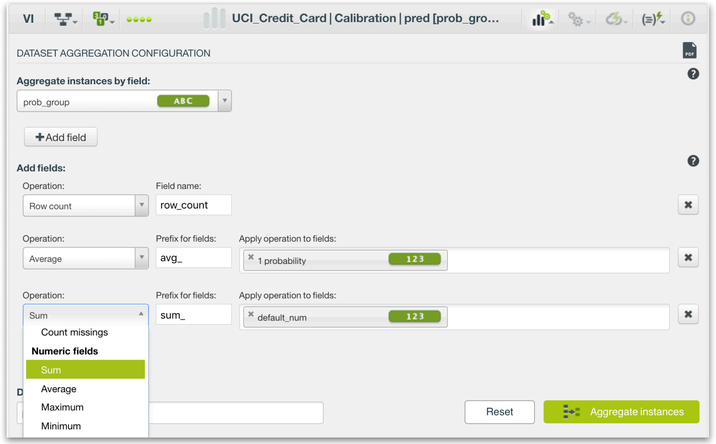
DATA PREPARATIONS: AGGREGATION BY GROUP

- To plot a reliability diagram in BigML we will need to proceed with several transformations
- Data needs to be aggregated over the 10 probability groups

INSTRUCTIONS:

- Let's aggregate by probability group. On the resulting dataset page, in the data transformations menu, select the “Aggregate instances” option (see screenshot 1)
- In the configuration panel select the prob_group field to aggregate on. We will need the row_count, the 1_probability average and the sum of default numeric value (see screenshot 2). Attention, on the operation selector several Average and Sum operations appear, scroll down for the regular numeric field operations
- The resulting dataset can be named prob_group_stats, it will have 10 rows, one for each group





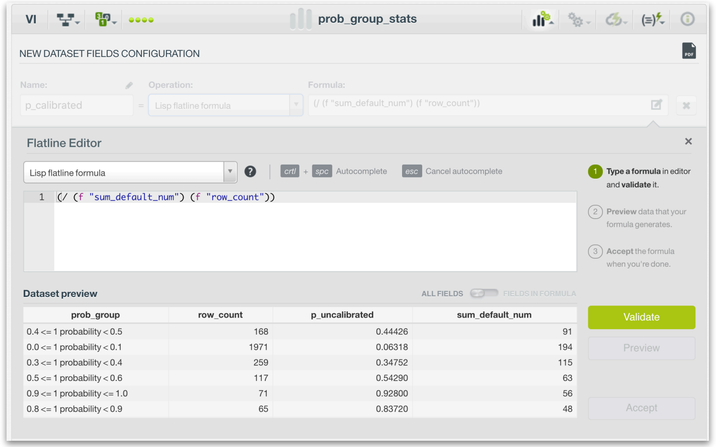
Reliability Diagram

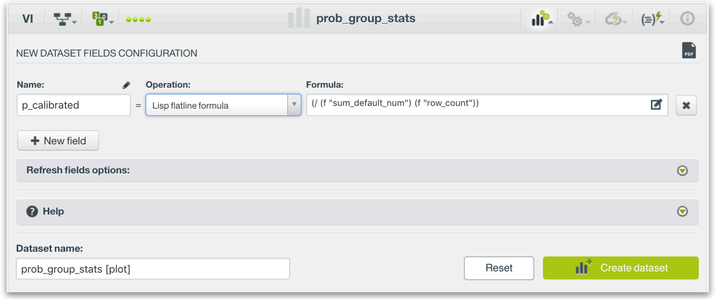
DATA PREPARATIONS: CALIBRATED PROBABILITY CALCULATION

- We have calculated the average predicted (uncalibrated) probability for each group
- It is now time to calculate the real (calibrated) probability average

INSTRUCTIONS:

- Rename the avg_1_probability field to p_uncalibrated (click on the pensil icon to edit)
- We should now add a new field dividing the number of default instances (sum_default_num) by the total amount of instances (row_count). This can be easily done using flatline, a simple programming language for such transformations. Flatline syntax is Lisp based, the web interface includes a compiler with validation and results preview features. Documentation is also available
- In the add fields configuration panel select the 2nd last option “Flatline Lisp Expression”, compile and preview the following code in the editor:
  `(/ (f "sum_default_num") (f "row_count"))`
- Name the new field p_calibrated

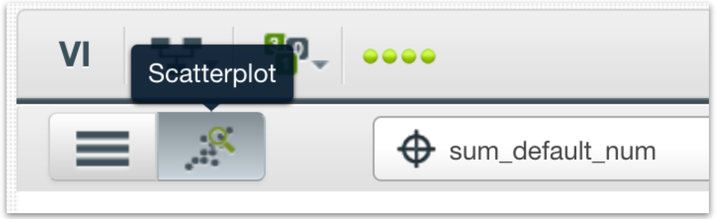




Reliability Diagram

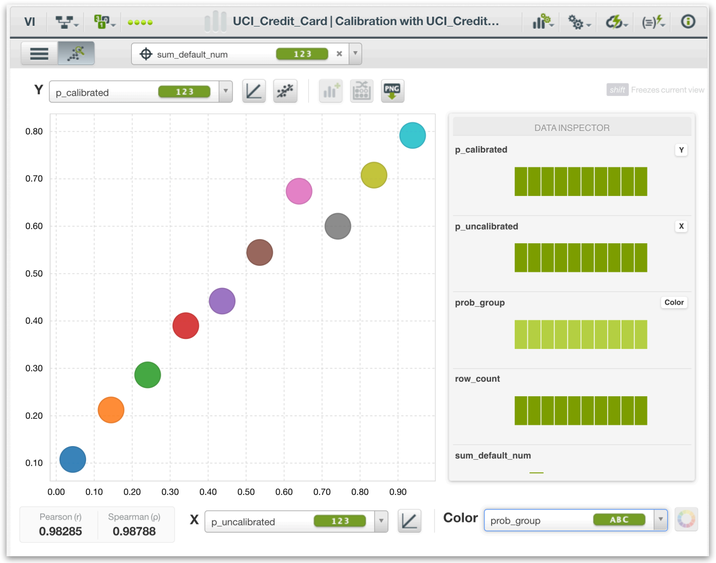
PLOT RELIABILITY DIAGRAM

- Data is ready to plot calibrated and uncalibrated probabilities by groups as a reliability diagram
- This diagram will help us understand if the ensemble previously created needs calibration
- If predictions probabilities are not close to real probabilities, calibration should be done



INSTRUCTIONS:

- On the resulting dataset click on the dynamic scatterplot view to plot
- Y = p_uncalibrated, X = p_calibrated, Color = any
- For a calibrated model plots would be near the diagonal line. If we imagine the diagonal line we can see a few measures are optimistic. Calibrating the model would then provide more realistic probabilities



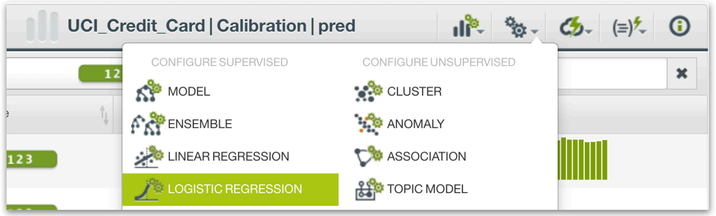
Plaat Scaling Calibration

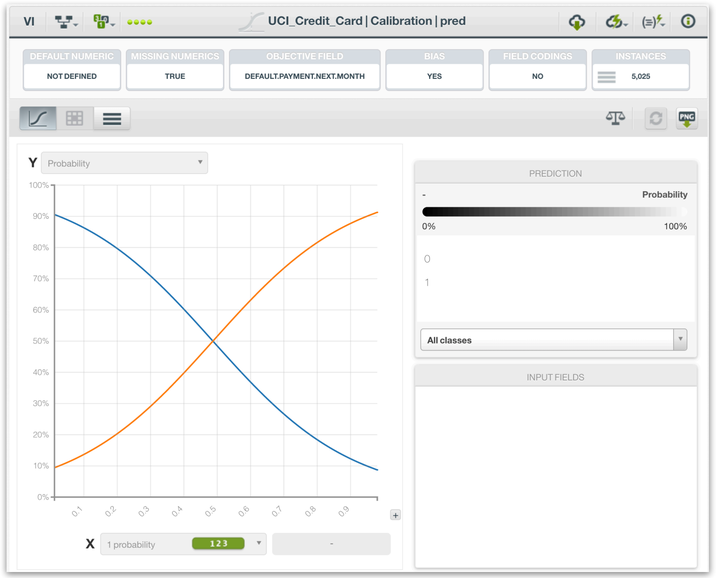
CALIBRATE THE ENSEMBLE

- Plaat scaling calibration consists in fitting a logistic regression over the model probabilites. The resulting probability for each instance will be calibrated

INSTRUCTIONS:

- On the previous batch prediction resulting dataset page, select the Logistic Regression option in the configurations menu, the configuration panel will appear
- We do not need to modify any parameters, only the attributes and objective field. The objective field needs to be set to default.payment.next.month
- On the fields selector disable all fields except default_pred and field 1_probability fields. Train a logistic regression with 1 single predictor. Default_pred is the objective field
- Explore the logistic regression resulting page graphics. The probability curve and coefficients can be visualized to understand the calibration





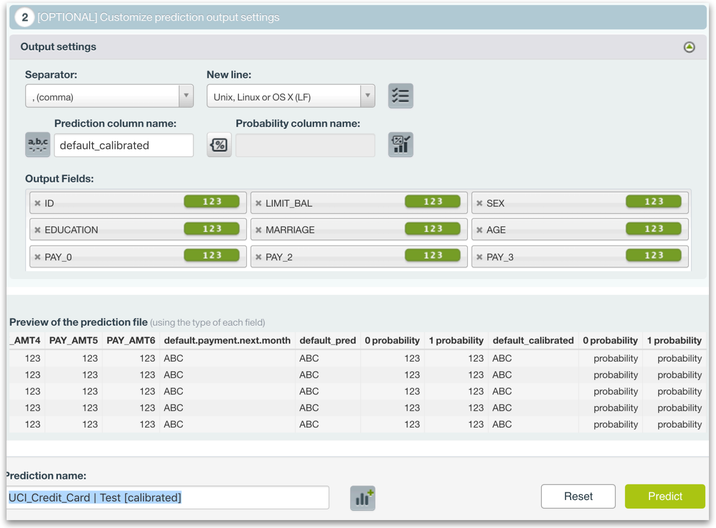
Test Results

PREDICT AND CALIBRATE ON THE TEST DATASET

- Let’s apply boosted ensemble predictions and calibrate resulting probabilities to evaluate test results

INSTRUCTIONS:

- On the boosted ensemble (find it in the supervised tab) apply a batch prediction over the test dataset, remember to configure the output as we did before (prediction result name + output probabilities) (see screenshot 1)
- On the logistic regression (find it in the supervised tab) apply a batch prediction over the resulting dataset above. Note the predictor field mapping is automatic as both should have the same name, otherwise it can be configured. The output needs to be configured so we have the resulting calibrated probabilities (see screenshot 2)
- Resulting dataset can be named UCI_Credit_Card | Test [calibrated] it will include final calibrated results. Rename the latest probability fields to calibrated_1_prob



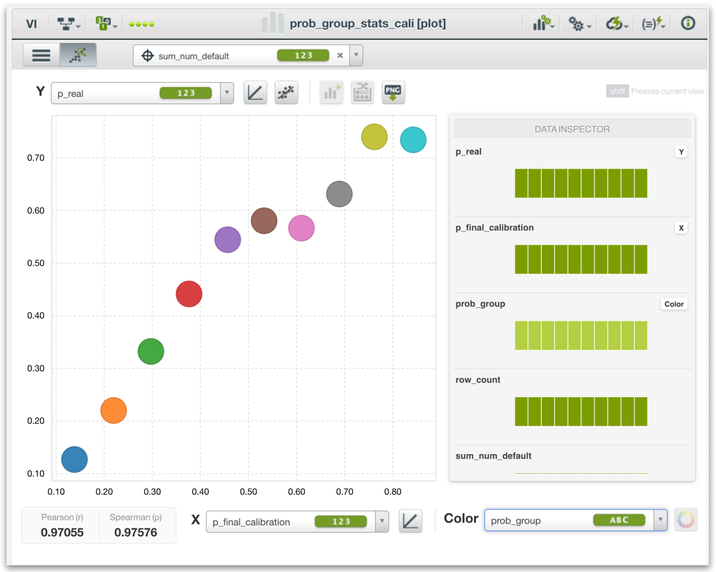
Results Reliability Diagram

PLOT THE RELIABILITY DIAGRAM WITH NEW PROBABILITIES

- Like we did before the new reliability diagram can be plot

INSTRUCTIONS:

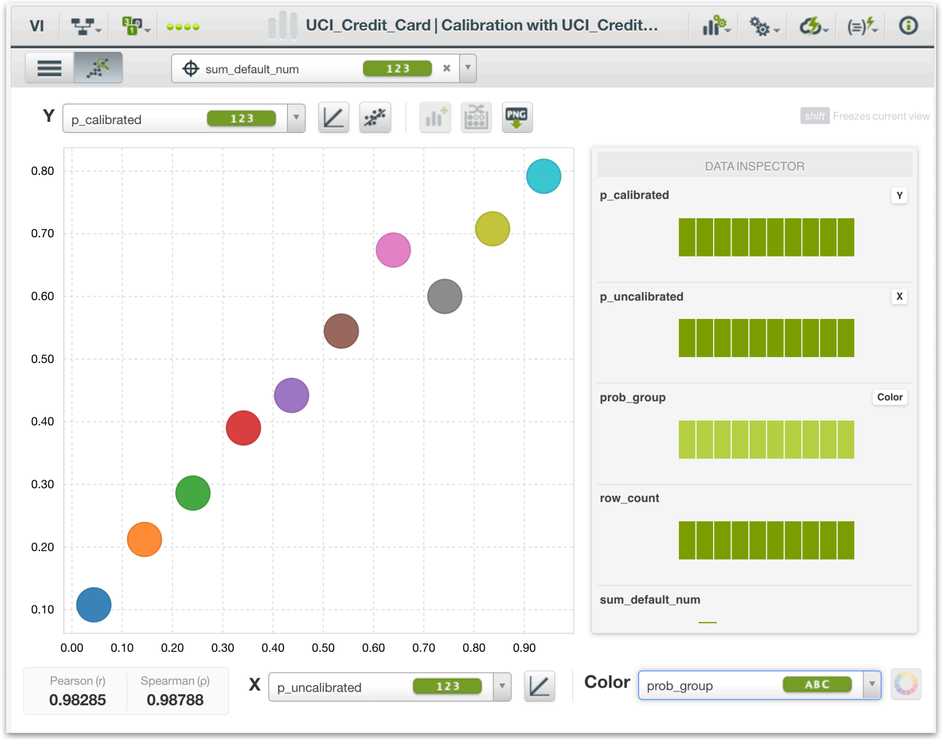
- Add the prob_group field over the calibrated probability together with the numeric type original default value
- Aggregate by prob_group. Keep the row_count and calculate the average calibrated probability together with the sum of defaults per each group. The average probability field can be named p_final_calibration
- Add the real average probability field with flatline (sum_default/ num_rows), it can be named p_real
- Plot the reliability diagram with the resulting dataset
- Is it closer to the diagonal virtual line than before?



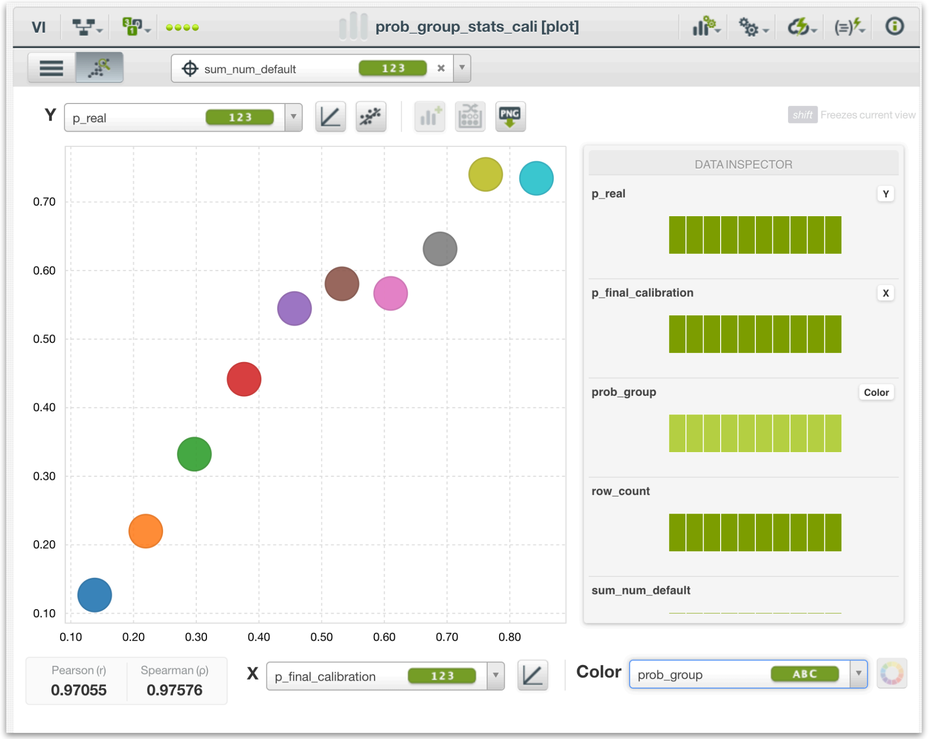
Reliability Diagrams Analysis

COMPARE RELIABILITY DIAGRAMS SIDE BY SIDE

- Let's see if there has been improvement in the probabilities after calibration
- Attention plot ranges change, let's use the virtual diagonal line as a reference in each case



Uncalibrated



Calibrated

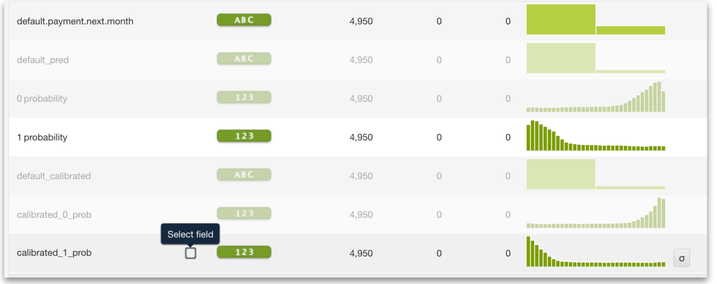
Rejection Curves Plot in Python

PLOT THE REJECTION CURVES IN PYTHON

- Rejection curves will provide a good overview to compare uncalibrated and calibrated probabilities

INSTRUCTIONS:

- Create a dataset with the fields we need. In the latest batch prediction dataset select the filter option. In the configuration panel exclude all fields except: default.payment.next.month, 1_probability (uncalibrated probability), calibrated_1_prob (calibrated probability)
- Export the dataset into a csv file. Open the dataset page and in the one-click menu select “Download CSV”. A csv file should appear in your downloads folder
- Edit the file and set the field names as follows: “truel.label", “uncalibrated”, “calibrated”
- Proceed with the python notebook provided to plot the curves



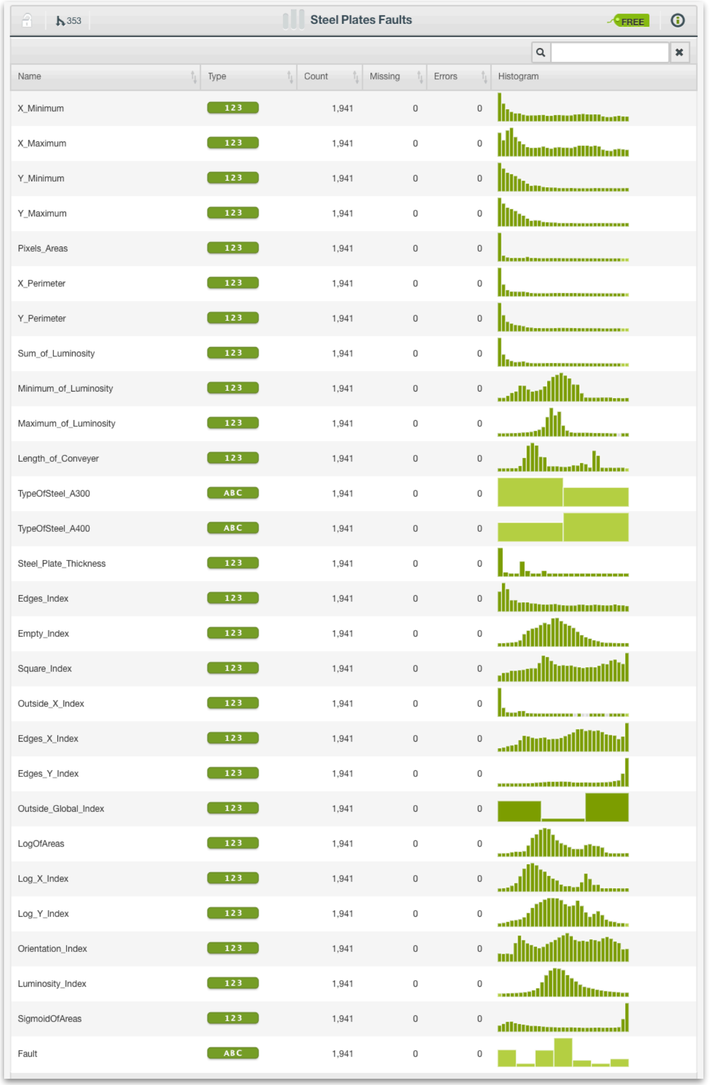
Agenda

1	Exercise 1: Calibration
2	Exercise 2: Novel Category Discovery
3	Exercise 3: Fraud Detection
4	Exercise 4: Data Cleaning

Exercise 2

STEEL PLATES FAULTS DATASET

- 1,941 instances, 242KB
- Each instance represents a faulty steel plate including several measures
- Fault is the objective field and has 7 possible values representing 7 different fault types
- The objective is to predict what fault type a given steel plate has



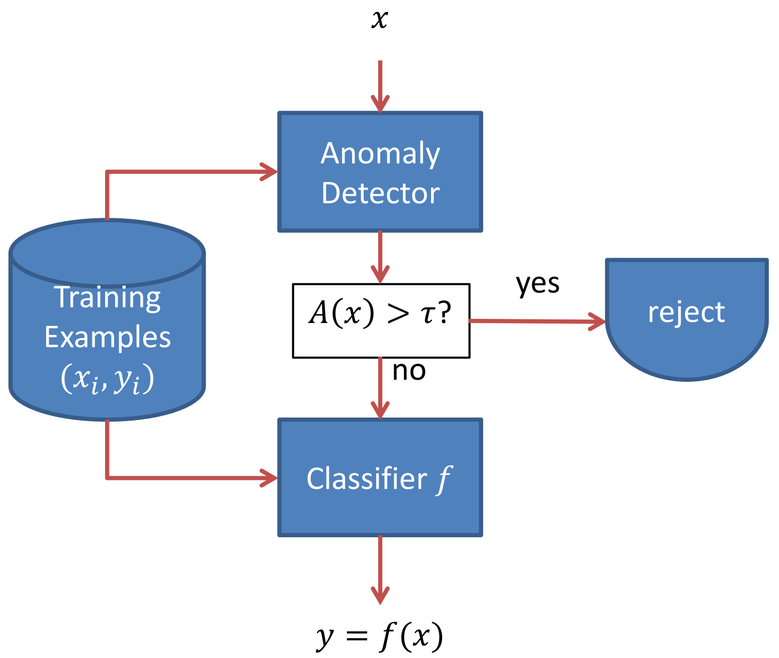
INSTRUCTIONS:

- Add the dataset into your account (link on image, click on free and clone)
- Explore dataset attributes and histograms. See the objective field histogram
- The dataset is already clean and Machine Learning ready

Exercise 2

STEP 1: NOVEL CATEGORIES DISCOVERY THROUGH ANOMALY DETECTION

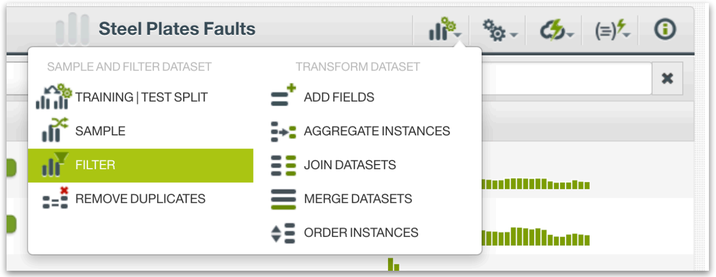
- Train a BigML anomaly detector (isolation forest) to determine whether or not a new data instance has a novel category
- Novel category instances should have higher anomaly scores than usual categories
- If a novel category is discovered it should be rejected



Data Preparation

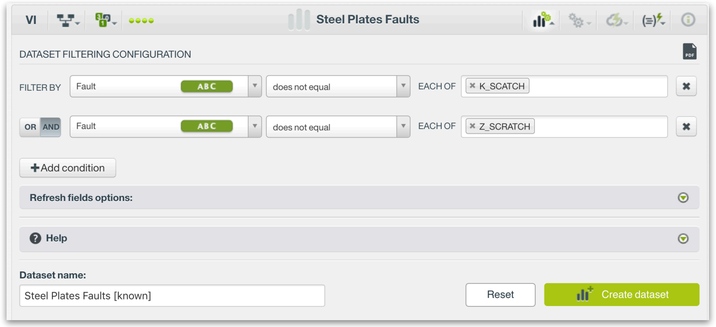
SEPARATE KNOWN AND NOVEL CATEGORIES DATA

- We will use K-Scratch and Z-Scratch as the two novel categories; all the remaining categories will be known in the training data
- First let's separate both kinds of instances into 2 different datasets



INSTRUCTIONS:

- In the datasets tab, open the previously imported dataset
- In the transformations menu (top right), select the filter option
- Set conditions to keep only known faults, this will create a new dataset
- Repeat the same keeping only novel faults into a second dataset



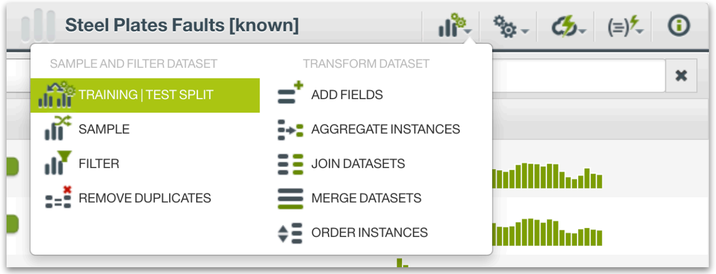
Data Preparation

PRODUCE TRAIN AND TEST DATASETS

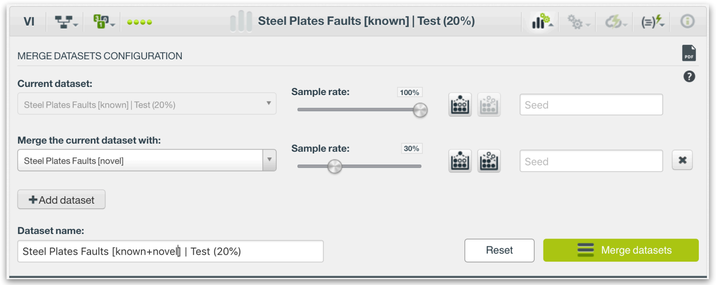
- Let’s create a training set with only known categories and a test set with both known and novel categories

INSTRUCTIONS:

- Split the known categories dataset into a training and test set. Open the dataset and in the transformations menu select the “Training Test Split” option. This will split the dataset randomly into 2 sets in the given proportion. 80/20 is fine



- The resulting training set will be used in the exercise. We still need to add novel categories into the test set. For this open the newly created test dataset and click on the “merge datasets” option in the transformation menu. Select the novel dataset created previously and sample to add 30% of novel category instances. This will produce a new test dataset including known and novel instances



Anomaly Detector

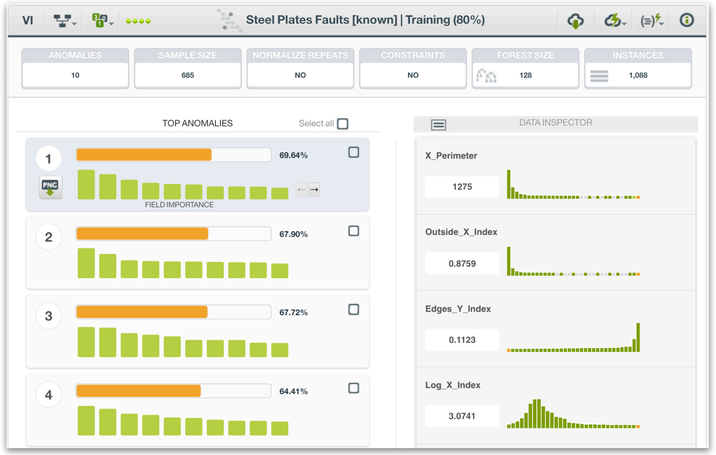
TRAIN ANOMALY DETECTOR

- Let’s train an anomaly detector with the training set previously prepared
- BigML anomaly detectors are implemented through isolation forest algorithm
- As it is trained over known data it should be able to discover novel categories

INSTRUCTIONS:

- Open the training dataset and in the one-click menu (cloud icon) select the “Anomaly” option. This will create an anomaly detector with the default parameters.

- Explore and understand the resulting anomaly detector page. On the left top 10 anomaly instances are displayed including their anomaly score and fields importance. When an anomaly is selected with the mouse its details appear on the right



Anomaly Detector Predictions

ANOMALY SCORE THE TEST DATASET

- We can now use the anomaly detector to score the test dataset
- We will then be able to plot results and determine if the novel discovery is working

INSTRUCTIONS:

- On the anomaly detector page, in the cloud menu click on the “Batch Anomaly Score” option. Select the test dataset previously generated and launch the anomaly score

- Take a look at the resulting dataset in BigML, a new score feature has been added representing the anomaly score of every instance. How does the score histogram look?

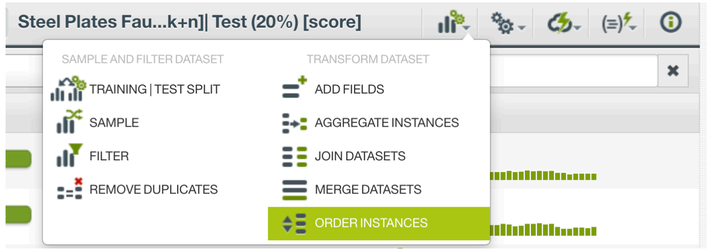
Plot Data Preparation

PREPARE ANOMALY SCORE DATA TO PLOT

- We should prepare data to plot and visualize the results
- We could plot the anomaly score together with the rejection percentage and color by known or novel
- Let's add fields into the score dataset in that sense

INSTRUCTIONS:

- On the score dataset page, in the transformations menu click on the "Order Dataset" option. Select the score field to order the data by score ascending. This will generate a new ordered dataset
- In the newly generated dataset we need to add new fields as follows
  - Percent rejected: now that the dataset is ordered we can simply get the percent of instances rejected by each score dividing the row number by the total amount of instances
  - Known vs Novel flag: we could generate a new field determining if the instance is novel or known based on it's fault value
  - This custom fields can be implemented using flatline, a BigML proprietary transformation language (see in next page)



Plot Data Preparation

ADD FIELDS WITH FLATLINE

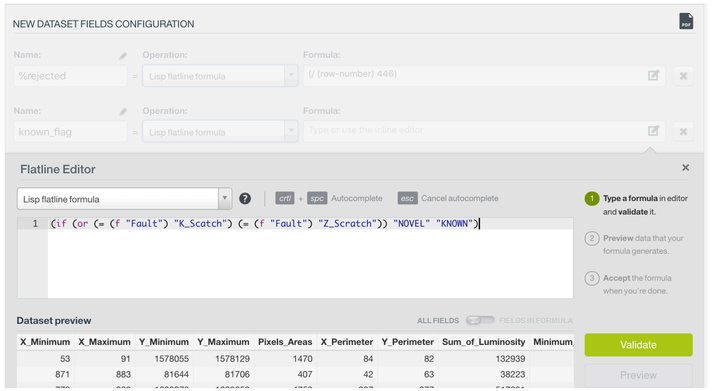
- Flatline is a simple programming language with similar to Lisp
- It allows adding fields with custom transformations
- BigML web interface includes a Flatline editor, compiler and validator
- Detailed documentation is available

INSTRUCTIONS:

- On the previously ordered dataset page, in the transformation menu select the “Add fields” option. Several add field methods are supported
- 2 fields should be added both using the method “Lisp Flatline Expression”, the second last. In both cases the editor can be opened by clicking on the edit icon
- “%rejected” field code: `(/ (row-number) 446)`
- “known_flag” field code:

```
(if (or (= (f "Fault") "K_Scatch") (= (f "Fault") "Z_Scratch")) "NOVEL" "KNOWN")
```

- A new dataset will be generated including both fields



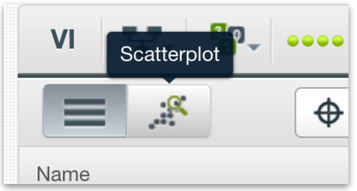
Plot Anomaly Scores

PLOT SCORES REJECTION PERCENTAGES

- The dynamic scatterplot functionality allows plotting data in 2 dimensions, sizes and coloring

INSTRUCTIONS:

- On the resulting dataset click on the dynamic scatterplot view to plot



- Y = anomaly score, X = % rejected, Color = known_flag

- How does it look? Are novel values related to high/low anomaly scores?

Exercise 2

STEP 2: NOVEL CATEGORY DISCOVERY THROUGH CONFIDENCE

- Train a supervised model to classify faults and calculate the probability difference between the first and second most probable classes
- High differences will represent confident classifications while small differences will refer to lower confidence in classification
- Plot such confidence scores to check if such metric is useful to discover new categories

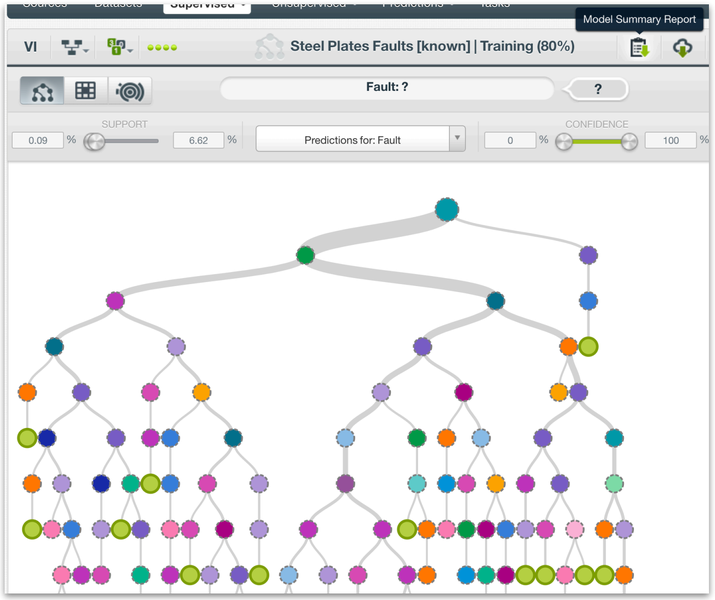
Model

TRAIN A DECISION TREE TO CLASSIFY FAULTS

- A supervised model can be trained to classify faults in the original dataset

INSTRUCTIONS:

- Open the previously used training Steel Plates Faults dataset and train a decision tree predicting the Fault. In the one-click menu click on the model option (BigML uses the last field as objective by default)



- The resulting decision tree will be displayed. Explore the tree and different visualizations. The tree main nodes and paths can be explored in the tree visualization while the sunburst shows an overall view with different coloring options. Finally the “Model Summary Report” button on the top right displays an interesting field importance summary (see image tooltip). Together with the Partial Dependence Plot (PDP) those 2 visualizations are available in most BigML supervised models

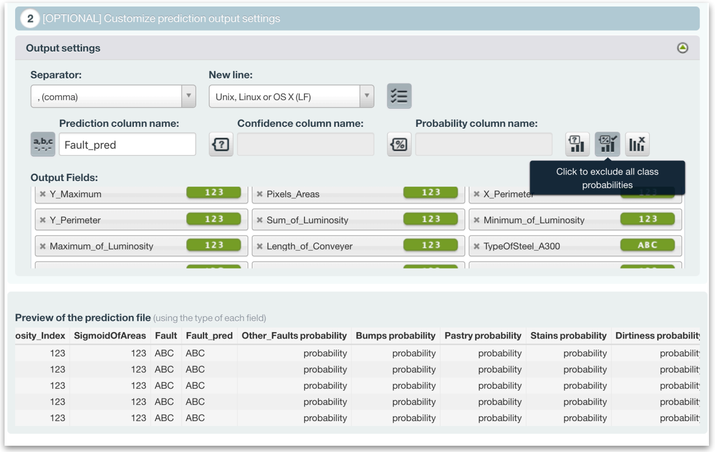
Batch Prediction

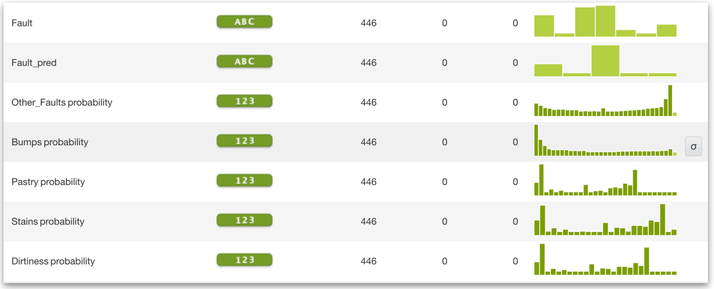
PROCESS A BATCH PREDICTION OVER THE TEST DATASET

- Current model classification predictions and class probabilities will be used to calculate the confidence values we need to plot

INSTRUCTIONS:

- In the previously generated model page, on the cloud menu select the “Batch Prediction” option
- Set the previously used test dataset as target for the batch prediction
- Configure the output:
  - The predicted field should be named “Fault_Pred”
  - The class probability option should be enabled
  - Note the resulting output format can be checked at the bottom, take a look before generating the batch prediction dataset
- Open the resulting dataset page, take a look at the resulting histograms





Confidence Metric

COMPUTE THE DIFFERENCE BETWEEN WINNER CLASSES PROBABILITY

- For each instance, the difference between the 2 top ranked classes will be used as a confidence metric
- High differences will determine a solid classification and low differences will correspond to low confidence

INSTRUCTIONS:

- From the batch prediction resulting dataset page select the “Add Fields” option in the transformations menu. Add a new field named “confidence_diff” with the following Lisp flatline code

```
(if
        (= (f "Other_Faults probability") (max (f "Other_Faults probability") (f "Bumps probability") (f "Pastry probability") (f "Stains probability") (f "Dirtiness probability")))
        (- (f "Other_Faults probability") (max (f "Bumps probability") (f "Pastry probability") (f "Stains probability") (f "Dirtiness probability")))
        (if
                (= (f "Bumps probability") (max (f "Other_Faults probability") (f "Bumps probability") (f "Pastry probability") (f "Stains probability") (f "Dirtiness probability")))
                (- (f "Bumps probability") (max (f "Other_Faults probability") (f "Pastry probability") (f "Stains probability") (f "Dirtiness probability")))
                (if
                        (= (f "Pastry probability") (max (f "Other_Faults probability") (f "Bumps probability") (f "Pastry probability") (f "Stains probability") (f "Dirtiness probability")))
                        (- (f "Pastry probability") (max (f "Other_Faults probability") (f "Bumps probability") (f "Stains probability") (f "Dirtiness probability")))
                        (if
                                (= (f "Stains probability") (max (f "Other_Faults probability") (f "Bumps probability") (f "Pastry probability") (f "Stains probability") (f "Dirtiness probability")))
                                (- (f "Stains probability") (max (f "Other_Faults probability") (f "Bumps probability") (f "Pastry probability") (f "Dirtiness probability")))
                                (- (f "Dirtiness probability") (max (f "Other_Faults probability") (f "Bumps probability") (f "Pastry probability") (f "Stains probability")))
                        )
                )
        )
)
```

- Inspect the resulting dataset field histogram

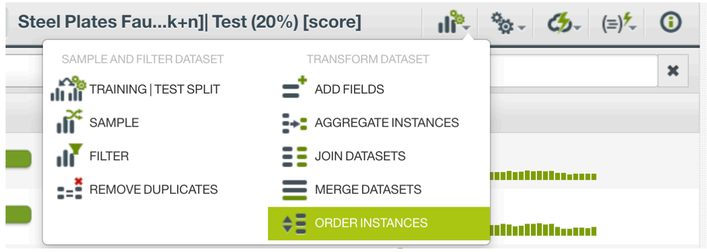
Plot Data Preparation

PREPARE CONFIDENCE DIFFERENCE DATA TO PLOT

- We should prepare data to plot and visualize the results
- We could plot the confidence difference together with the rejection percentage and color by known or novel
- Let's add fields into the score dataset in that sense, just like before

INSTRUCTIONS:

- On the resulting dataset page, in the transformations menu click on the "Order Dataset" option. Select the confidence field to order the data by ascending. This will generate a new ordered dataset
- In the newly generated dataset we need to add new fields with flatline like we did before (find instructions in previous slides)
  - Percent rejected: now that the dataset is ordered we can simply get the percent of instances rejected by each score dividing the row number by the total amount of instances
  - Generate a new field by using if statements to check whether the fault value is one of the novel fault types



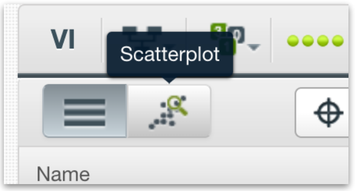
Plot Anomaly Scores

PLOT CONFIDENCE WITH REJECTION PERCENTAGES

- The dynamic scatterplot functionality allows plotting data in 2 dimensions, sizes and coloring

INSTRUCTIONS:

- On the resulting dataset click on the dynamic scatterplot view to plot



- Y = confidence difference, X = % rejected, Color = known_flag

- How does it look? Are novel values related to high/low confidences?

Exercise 2

STEP 3: DETECTING NOVEL CLASSES USING AN OLD VS. NEW CLASSIFIER

- Train a supervised model to classify novel vs known instances without using the class labels
- Via a K-fold cross validation produce respective batch predictions to the evaluated folds
- Merge resulting batch predictions and plot novel classes probabilities to inspect resulting data

Data Preparation

SEPARATE DATA AND SPLIT INTO FOLDS

- Using the dataset with the known vs novel flag we would normally perform 10-fold cross-validation
- We will manually perform 2-fold cross validation

INSTRUCTIONS:

- Split the original dataset with the known vs novel flag into 2 equal parts. Click on the “Training |Test Split” option in the transformations menu and select 50% ratio. Suffix the resulting dataset names 1 and 2 rather than train and test
- This will generate 2 datasets

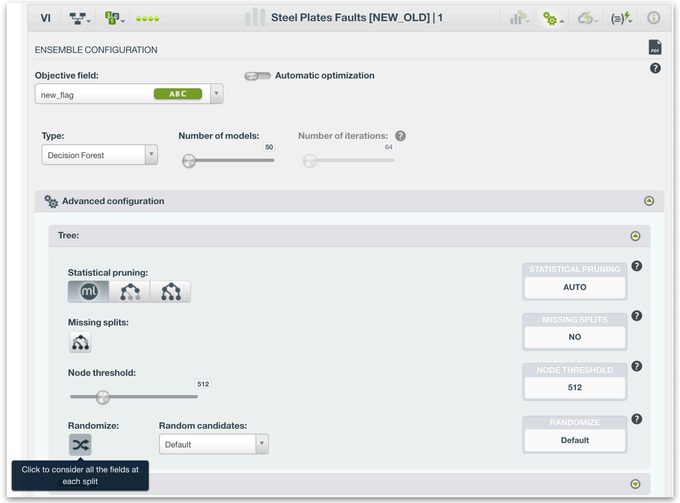
Random Forests

TRAIN 2 RANDOM FOREST ENSEMBLES

- From the 2 datasets 2 random forest models need to be trained for further mutual batch predictions

INSTRUCTIONS:

- For each dataset, in the gears menu within its page, click on the “Ensemble” option. This will open the Ensemble configuration menu. This algorithm allows combining decision trees in different ways emulating bagging decision forests, random decision forests or boosted trees



- The decision forest option is selected by default. Set the number of trees to 50. Click on the “Randomize” button to emulate a Random Decision Forest

- 2 ensembles will be created, explore the graphical interfaces

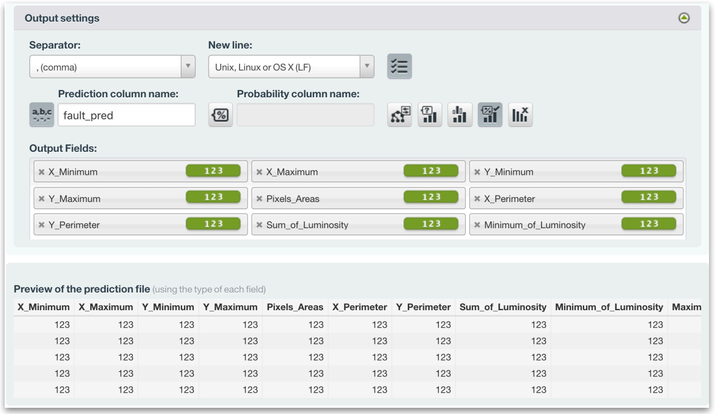
Batch Predictions

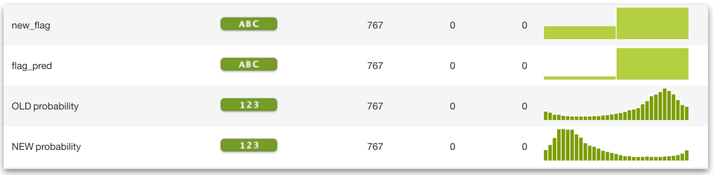
BOTH RANDOM FOREST MODELS BATCH PREDICTIONS

- Batch predictions need to be processed for both models with their respective opposite datasets
- Resulting prediction datasets need to be merged into a single dataset for further global analysis

INSTRUCTIONS:

- In both previously generated ensemble pages, on the cloud menu select the “Batch Prediction” option
- Set the opposite respective dataset for each ensemble, meaning random forest 1 batch prediction will be done against dataset 2 and vice versa
- Configure the output:
  - The predicted field should be named “Fault_Pred”
  - The class probability option should be enabled
  - Note the resulting output format can be checked at the bottom, take a look before generating the batch prediction dataset
- Merge both resulting datasets into one (same prediction field names recommended)





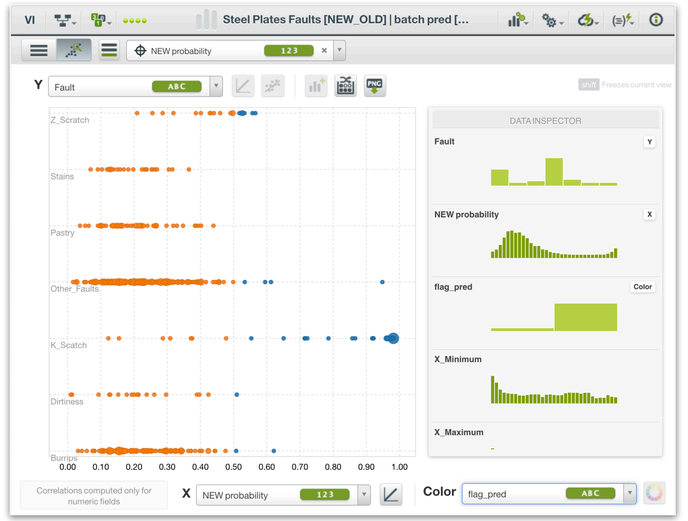
Plot Probabilities

PLOT NOVEL CLASS PROBABILITIES

- The dynamic scatterplot functionality allows plotting data in 2 dimensions, sizes and coloring

INSTRUCTIONS:

- On the resulting dataset click on the dynamic scatterplot view to plot
- Y = Fault, X = NEW Probability, Color = flag_pred
- How does it look? Are novel values related to high/low probability?



Agenda

1	Exercise 1: Calibration
2	Exercise 2: Novel Category Discovery
3	Exercise 3: Fraud Detection
4	Exercise 4: Data Cleaning

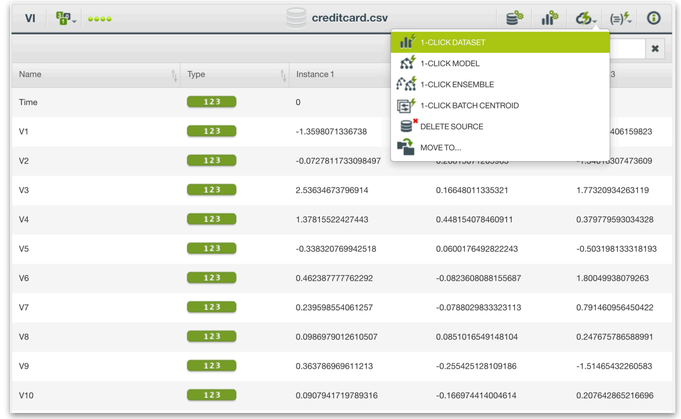
Exercise 3

BANK TRANSACTIONS FRAUD DETECTION DATASET

- Original dataset found in Kaggle is bigger, a random sample has been used for performance reasons
- 95,662 credit card account transactions instances 4.3MB, 31 attributes
- Most attributes are numeric values and cannot be interpreted, probably due to a previous PCA
- Class is the objective field and has 0,1 values. 1 means fraud. The goal is predicting what transactions are fraudulent

INSTRUCTIONS:

- To import the data, drag and drop the creditcards_small.csv file provided into your BigML account, a new source will be created
- Find the new source in the sources tab, open it, check it looks good
- Generate a dataset through the 1-click-dataset option in the cloud menu



Exercise 3

STEPS

- Train an anomaly detector over the whole dataset without considering the class result
- Simulate fraud analyst tasks by checking the top anomalies one by one
- Verify if the resulting top anomalies correspond to fraud transactions
- Visualize results

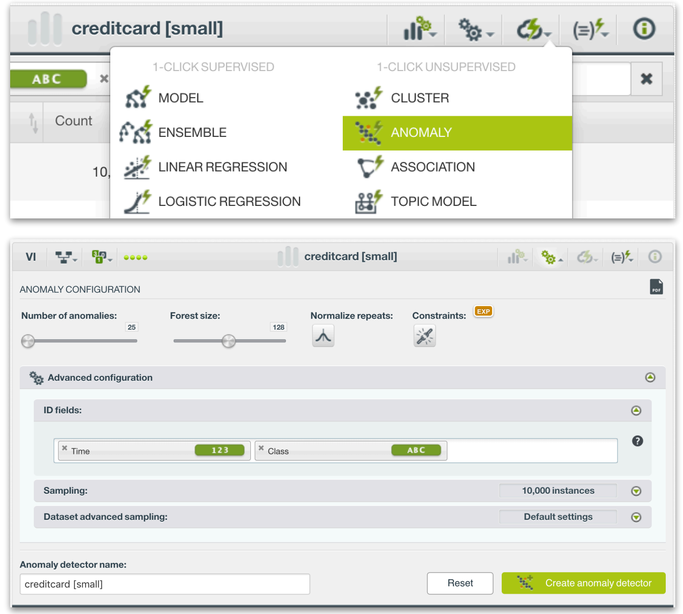
Anomaly Detector

DETECTING CREDIT CARDS FRAUD WITH AN ANOMALY DETECTOR

- Our dataset is too unbalanced to use a classifier for this problem
- With such unbalanced datasets anomaly detectors can be used to classify
- Will an anomaly detector be helpful to find fraudulent transactions in this case?

INSTRUCTIONS:

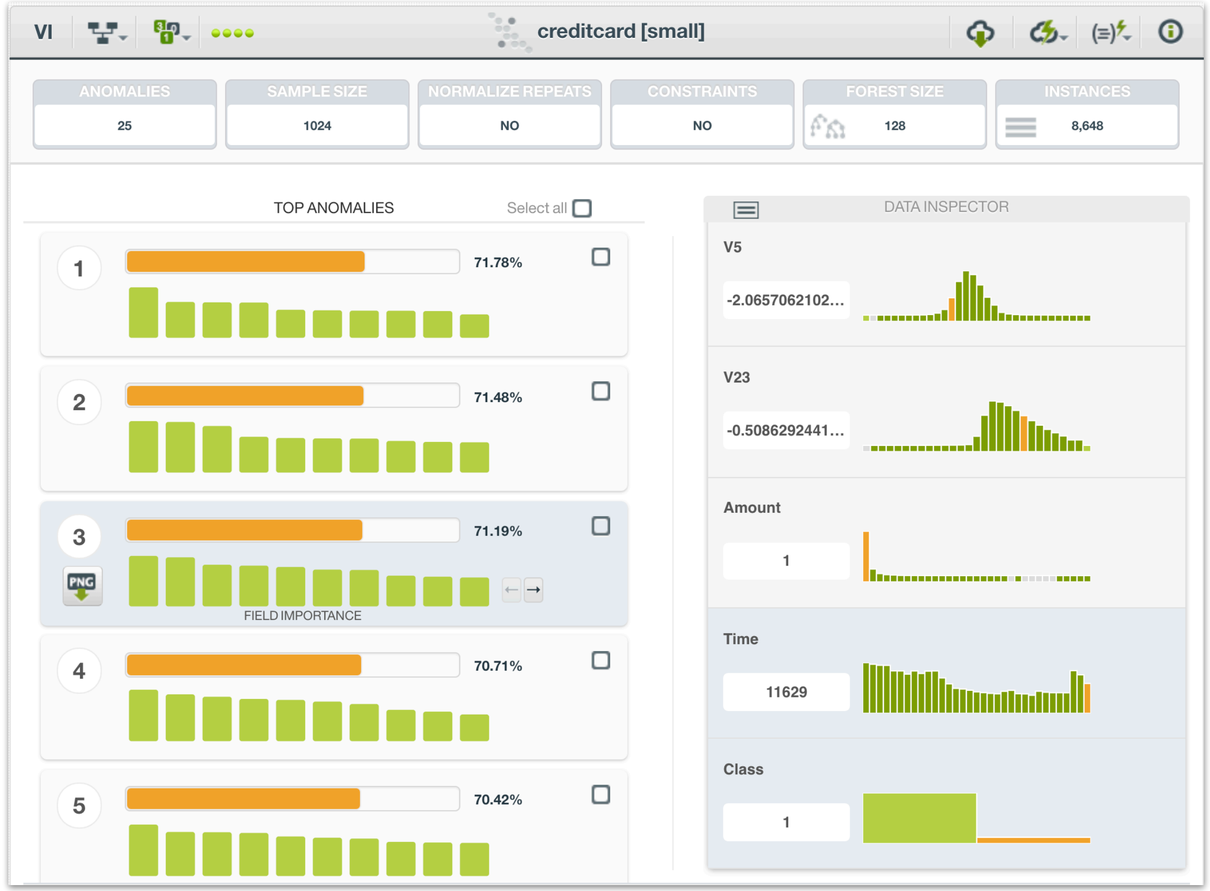
- From the newly created database page, in the gears menu, click on the “Anomaly” option to configure an anomaly detector
- In the configuration page select 25 anomalies to be displayed, set the class and Time fields as ID fields so they are not taken into account in the isolation forest but still displayed in the results
- Let’s imagine you’re a transactions fraud analyst. Explore the resulting anomalies page, the 25 top anomalies are displayed on the left screen side and details of each instance can be explored in the right part



Anomaly Detector

ANALYSING ANOMALIES

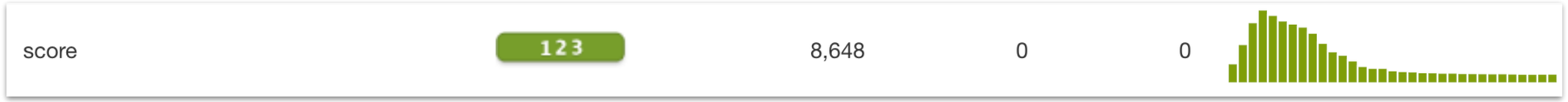
- A fraud analyst would be given a list of suspicious transactions
- How many of the top 25 anomaly instances are actually fraud?



Batch Anomaly Score

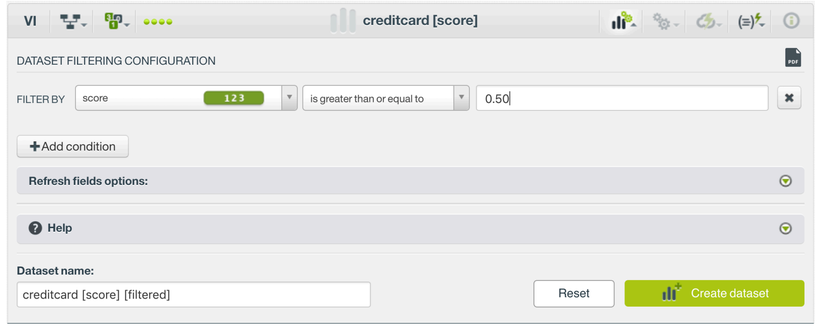
EXTRACT ANOMALY SCORES

- Let's extract the anomaly scores for all instances so we can plot and visualize results



INSTRUCTIONS:

- From the anomaly detector page launch a batch anomaly score into the original dataset
- This will generate a new dataset with an anomaly score assigned to every instance. Take a look at the score histogram, how does it look?
- We would like to plot the top anomalies. Let's filter by anomaly score. In the transformations menu select the filter option and keep all instances with a score greater than 0.5



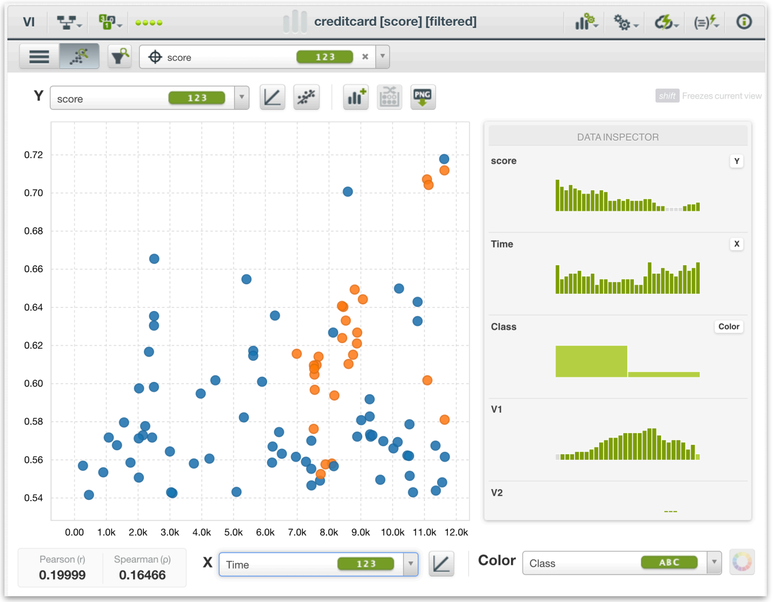
Plot Results

PLOT ANOMALY SCORES AND CLASSES TO VERIFY RESULTS

- Let's plot fraud detection anomaly score results from the resulting dataset

INSTRUCTIONS:

- On the resulting dataset click on the dynamic scatterplot view to plot
- Y = score, X = Time, Color = Class
- How does it look? Is the anomaly detector finding fraud cases over 0.5? Is there a better score threshold to optimize the classifier?



Agenda

1	Exercise 1: Calibration
2	Exercise 2: Novel Category Discovery
3	Exercise 3: Fraud Detection
4	Exercise 4: Data Cleaning

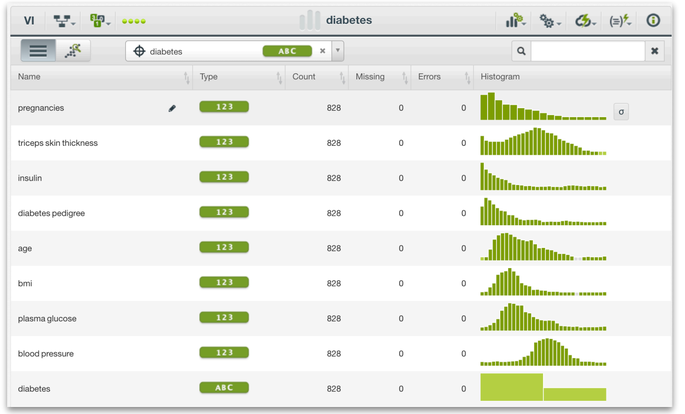
Exercise 4

DIABETES DATASET

- 828 instances each representing a medical patient
- 8 features and a boolean objetive field determining whether or not the patient is diabetic
- The goal is predicting diabetes diagnostic over new patients

INSTRUCTIONS:

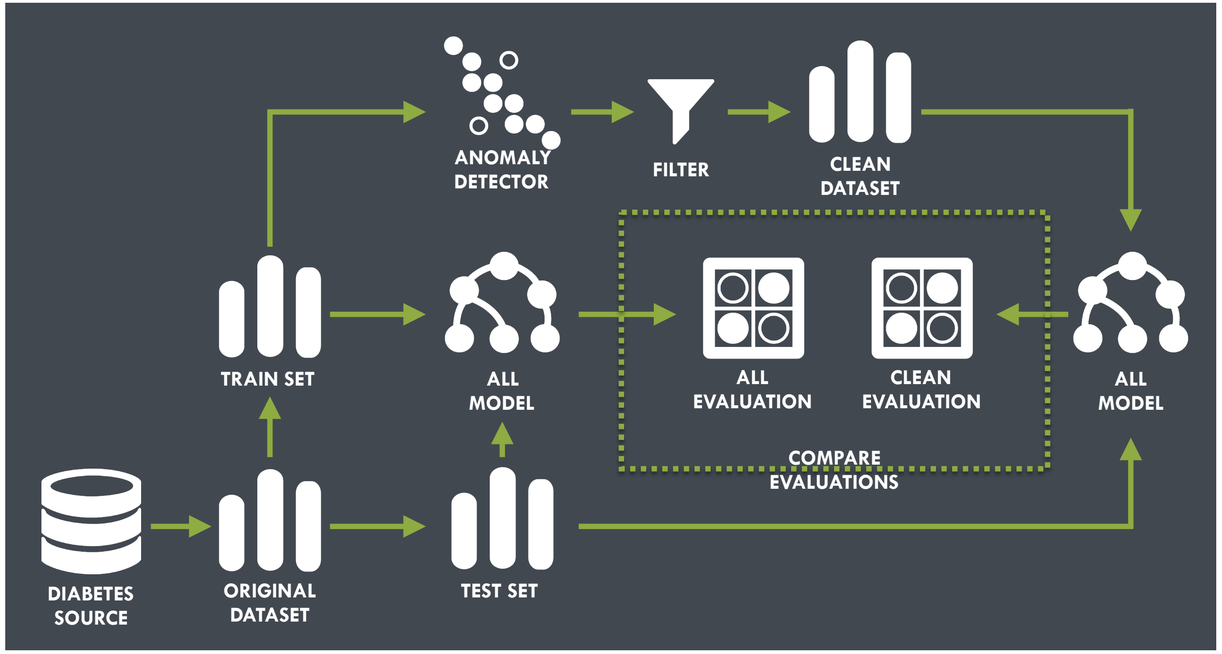
- To import the data, drag and drop the diabetes.csv file provided into your BigML account, a new source will be created
- Find the new source in the sources tab, open it, check it looks good
- Generate a dataset through the 1-click-dataset option in the cloud menu



Exercise 4

STEPS

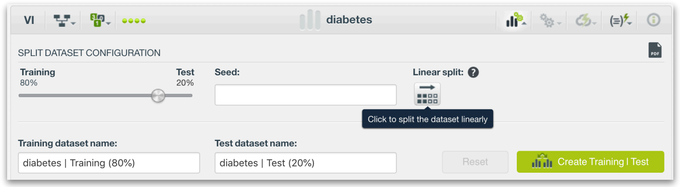
- Train and evaluate a supervised model to predict diabetes
- Analyse training data top anomalies
- Filter out top anomalies to clean and train an identical supervised model with the clean training set
- Use the test set to evaluate both supervised models
- Analyse and see if cleaning helped improve the model

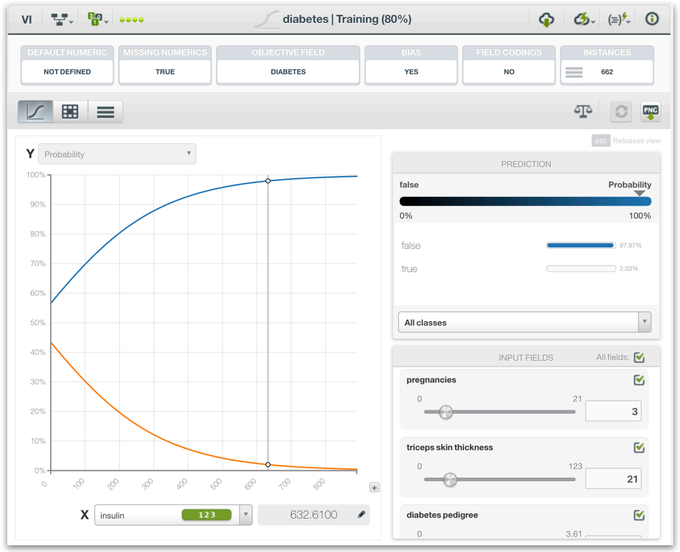


Logistic Regression

SPLIT DATA AND TRAIN A LOGISTIC REGRESSION

- Logistic regression is the classification algorithm we are going to use in this case
- We’re going to train a logistic regression after splitting the dataset





- Split the dataset (80/20). In the split menu select the Linear option so it is not split randomly (see screenshot)
- On the training dataset, train a Logistic Regression supervised model with the default parameters
- Explore the Logistic Regression page. Probabilities of each variable are plot and other variable values can be modified. LR coefficients are available as well such as the PDP visualization

Logistic Regression Evaluation

EVALUATE THE LOGISTIC REGRESSION MODEL

- Let's evaluate this classification

INSTRUCTIONS:

- On the logistic regression cloud menu select the “Evaluation” option. A dialog will appear to select the dataset, the corresponding test dataset needs to be picked if it is not already
- The resulting classification evaluation page will appear including the confusion matrix and multiple metrics
- Set the positive class as “true" in the selector
- Extend the evaluation view to display the ROC curve
- Explore the evaluation details

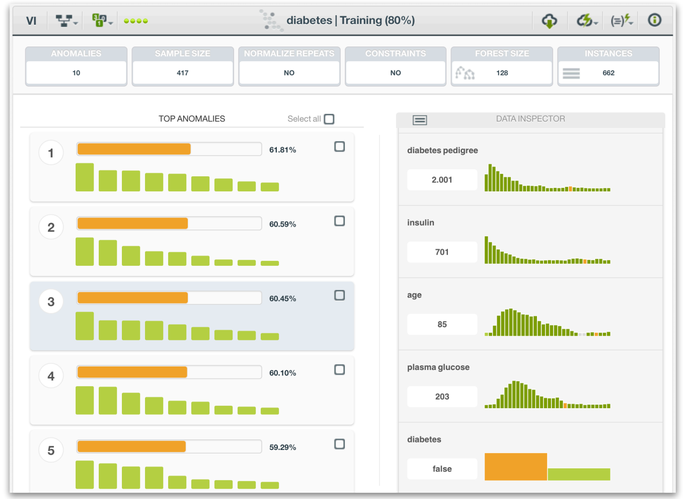
Clean Training Dataset Outliers

TRAIN AN ANOMALY DETECTION

- Let’s train an anomaly detector with the training data and analyse possible outliers

INSTRUCTIONS:

- On the training dataset page train an anomaly detector by clicking on the one-click anomaly default option
- Analyse the resulting top anomalies. Is there anything strange?



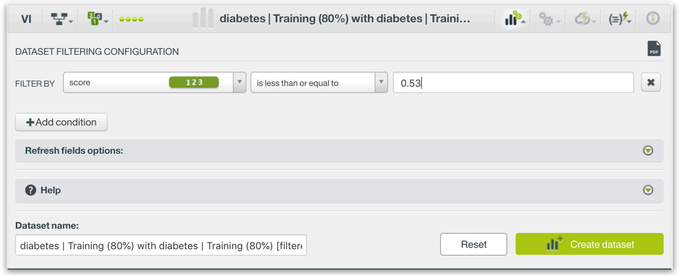
Clean Training Dataset Outliers

REMOVE TOP ANOMALIES

- Let's filter out top anomalies from the training set to remove outliers

INSTRUCTIONS:

- On the anomaly detector generate a batch anomaly score over the original training dataset
- Inspect the resulting histogram
- Select the filter functionality and keep instances with score under 53%
- Name the resulting dataset with the clean suffix



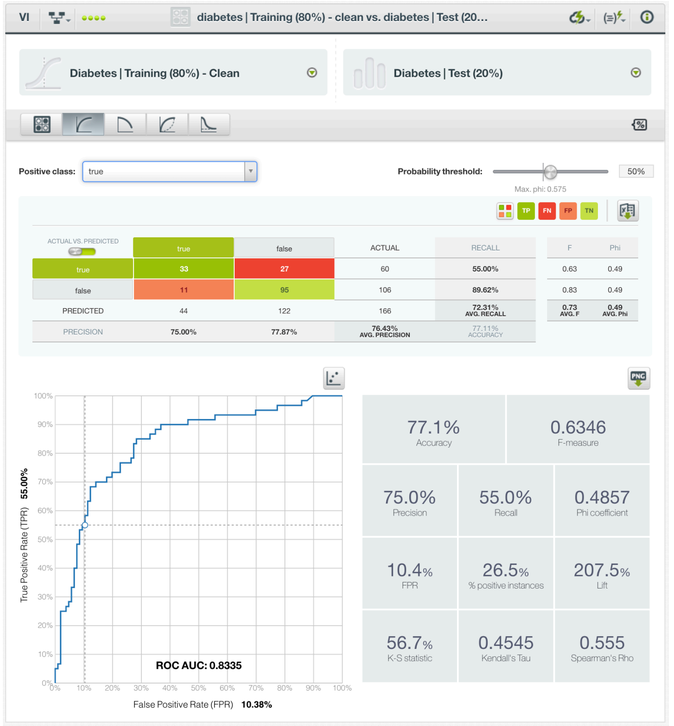
Clean Evaluation

TRAIN AND EVALUATE LOGISTIC REGRESSION WITH THE CLEAN DATA

- Let's train another logistic regression with the clean data

INSTRUCTIONS:

- On the resulting dataset train a new Logistic Regression. In the configuration menu select the Logistic regression option, define the diabetes field as the objective field an d disable the score field
- Inspect the results and evaluate it with the original test dataset
- Inspect the evaluation. Is the ROC AUC better than before? Is the accuracy better than before?



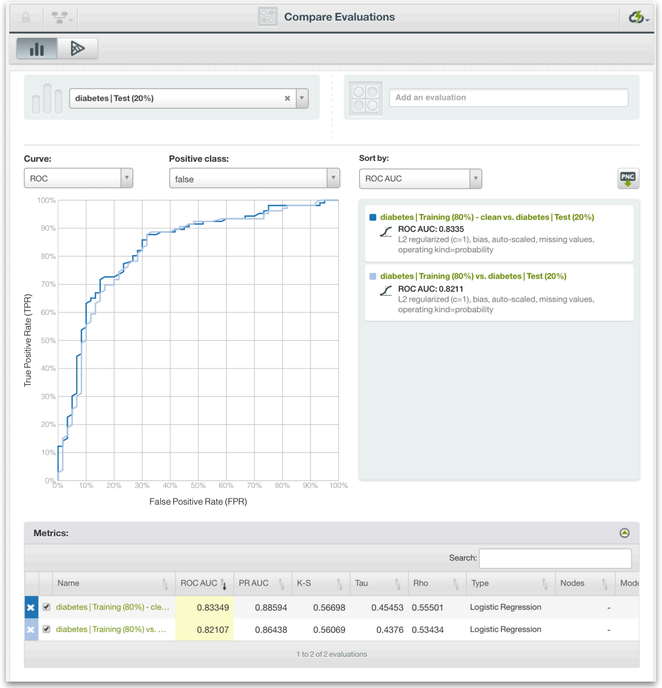
Compare Evaluations

COMPARE ORIGINAL AND CLEAN EVALUATIONS

- Let's visualize original and clean evaluations together in the comparison screen

INSTRUCTIONS:

- On the clean evaluation page, in the cloud menu select the option “Compare Multiple Evaluations”. The comparison screen will be displayed
- In the evaluation selector pick the original evaluation (type anything such "diabetes" and available evaluations will display, then select the original one)
- Visualise both curves and metrics together see in what bits of the curve the clean data is working better




DSN
Data Science Nigeria


1868
Oregon State
University


bigml®