Introduction to HPCC Systems, ECL, and ECL Machine Learning

Dan S. Camper (Sr. Architect) and Lili Xu (Software Engineer III)
HPCC Solutions Lab

January 17, 2019

LexisNexis® RISK SOLUTIONS

Agenda

About LexisNexis Risk Solutions

We believe in the power of data and advanced analytics for better risk management.

Data, Analytics and Technology

LexisNexis Risk Solutions leverages its industry-leading Big Data computing platform with vast data assets and a proprietary fast-linking technology to enable businesses of all sizes to better analyze and understand data at scale, improving time-to-results and decisions.

Solutions We Provide

With our solutions, our customers transform their risk decision making and are empowered to make better decisions easier. We help them with business challenges like fighting fraud, facilitating compliance, streamlining workflows and increasing efficiencies, improving health outcomes and keeping communities safe by providing timely insights for business decisions.

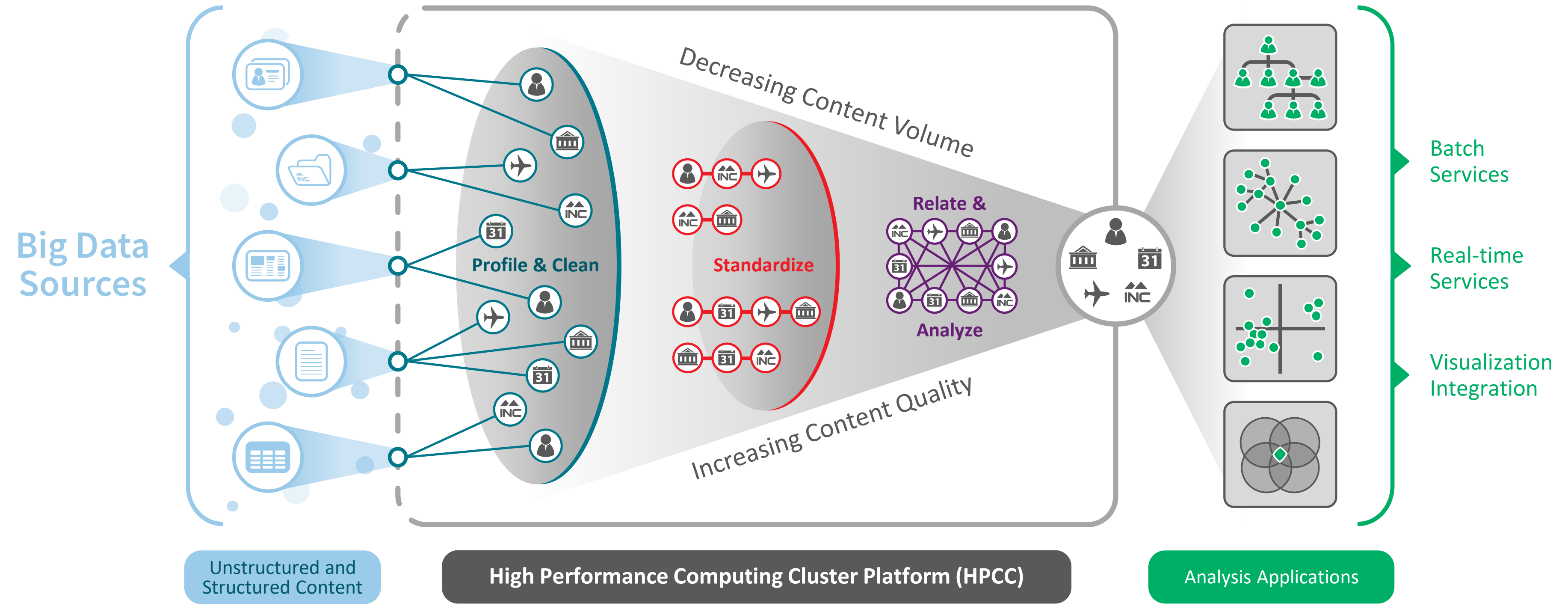
Markets We Serve

LexisNexis Risk Solutions provides solutions across multiple industries, including Insurance, Financial Services, Collections and Recovery, Retail, Health Care and Communications. We also work with all levels of local, state, and federal governments and their agencies. We serve customers in more than 100 countries.

LexisNexis® RISK SOLUTIONS

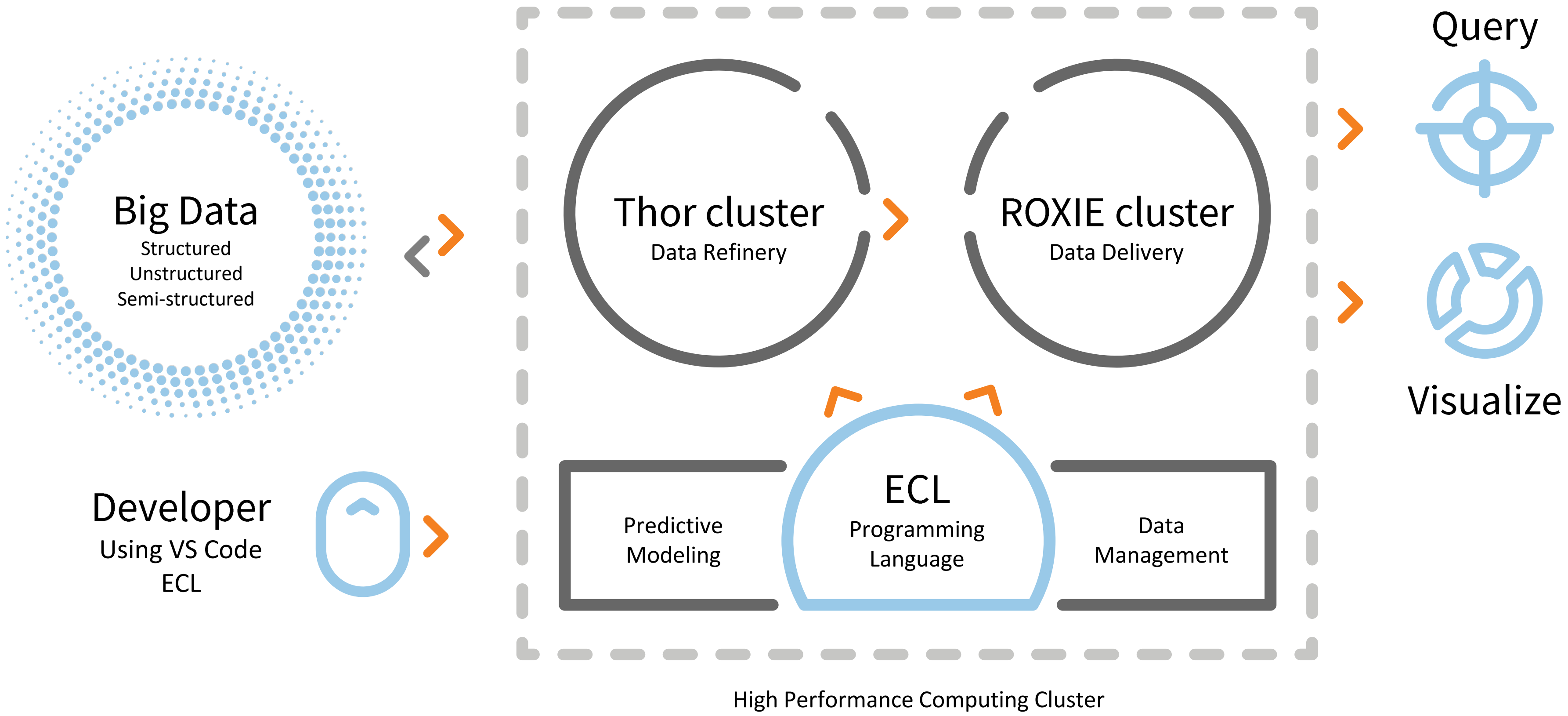
HPCC Systems (Small to Big Data) ETL

Integrate > Profile > Clean > Standardize > Analyze > Deliver

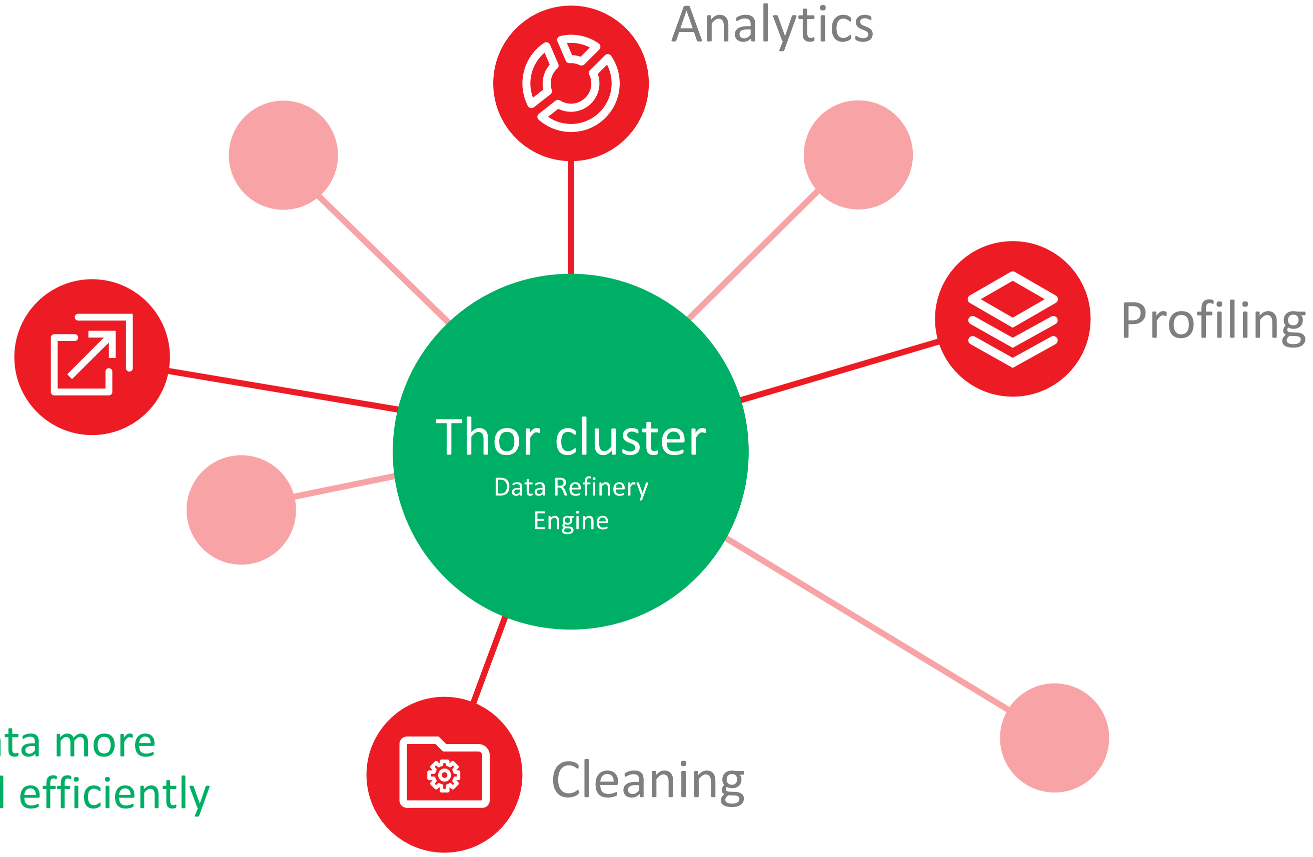

Big Data Sources
Decreasing Content Volume
Profile & Clean
Standardize
Relate & Analyze
Increasing Content Quality
Batch Services
Real-time Services
Visualization Integration
Unstructured and Structured Content
High Performance Computing Cluster Platform (HPCC)
Analysis Applications

LexisNexis® RISK SOLUTIONS

The HPCC Systems Components

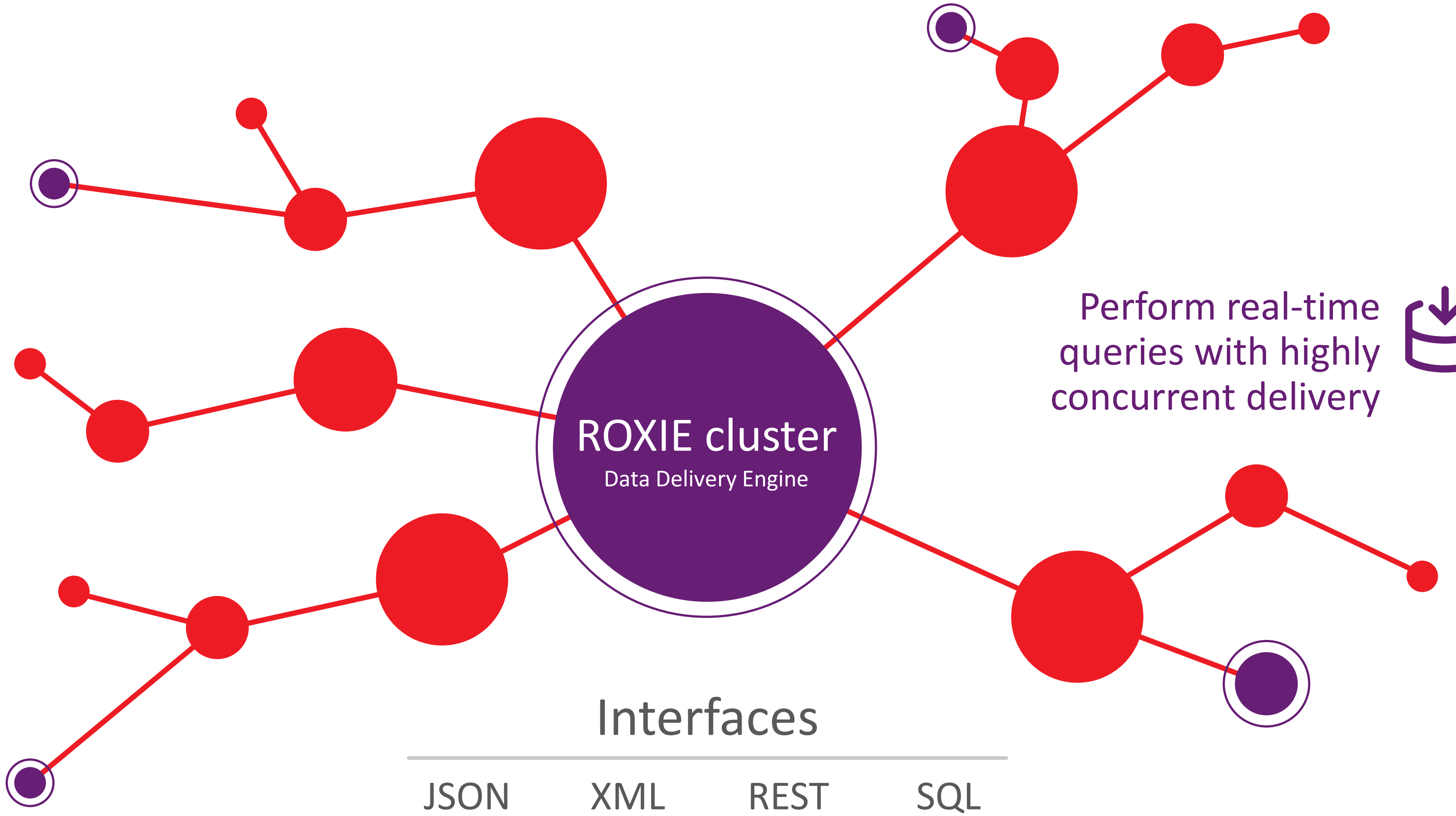


An Intro to Thor



Manage data more quickly and efficiently

An Intro to ROXIE



Perform real-time queries with highly concurrent delivery

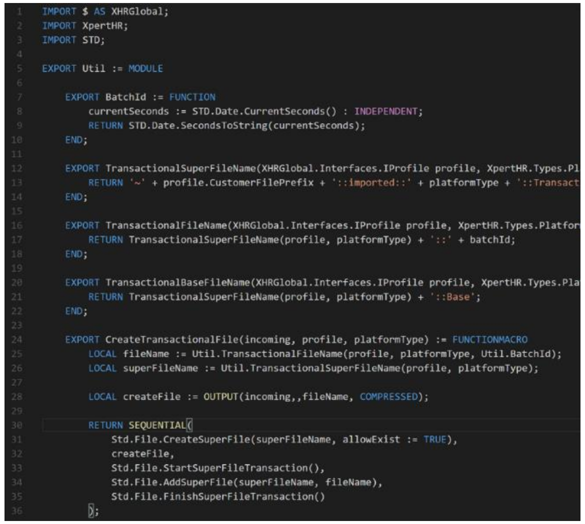
Interfaces

JSON XML REST SQL

An Intro to ECL

ECL

Enterprise Control Language



```
IMPORT $ AS XHRGlobal;
IMPORT XpertHR;
IMPORT STD;

EXPORT Util := MODULE

    EXPORT BatchId := FUNCTION
        currentSeconds := STD.Date.CurrentSeconds() : INDEPENDENT;
        RETURN STD.Date.SecondsToString(currentSeconds);
    END;

    EXPORT TransactionalSuperFileName(XHRGlobal.Interfaces.IProfile profile, XpertHR.Types.Pl
        RETURN '~' + profile.CustomerFilePrefix + '::imported::' + platformType + '::Transact
    END;

    EXPORT TransactionalFileName(XHRGlobal.Interfaces.IProfile profile, XpertHR.Types.Platfor
        RETURN TransactionalSuperFileName(profile, platformType) + '::' + batchId;
    END;

    EXPORT TransactionalBaseFileName(XHRGlobal.Interfaces.IProfile profile, XpertHR.Types.Pla
        RETURN TransactionalSuperFileName(profile, platformType) + '::Base';
    END;

    EXPORT CreateTransactionalFile(incoming, profile, platformType) := FUNCTIONMACRO
        LOCAL fileName := Util.TransactionalFileName(profile, platformType, Util.BatchId);
        LOCAL superFileName := Util.TransactionalSuperFileName(profile, platformType);

        LOCAL createFile := OUTPUT(incoming,,fileName, COMPRESSED);

        RETURN SEQUENTIAL(
            Std.File.CreateSuperFile(superFileName, allowExist := TRUE),
            createFile,
            Std.File.StartSuperFileTransaction(),
            Std.File.AddSuperFile(superFileName, fileName),
            Std.File.FinishSuperFileTransaction()
        );
```

Powerful language built for big data

How to do it ✗ vs. ✓ What to do

The ECL Programming Language

- Designed to allow data operations to be specified in a manner which is easy to optimize and parallelize
- Declarative in nature (“what you want done, rather than how to do it”)
- Extremely succinct
- Data centric, extensible and internally abstract
    - Minimizes data movement
    - Does work once
    - Keeps all nodes equally busy
- Implicitly parallel

- Compared to MapReduce:
    - MAP -> PROJECT/TRANSFORM
    - SHUFFLE (phase 1) -> DISTRIBUTE(,HASH(KeyValue))
    - SHUFFLE (phase 2) -> SORT(,LOCAL)
    - REDUCE -> ROLLUP(,Key,LOCAL)

Return list of actors appearing most often in movies

```
FO := IMDB.File_actors;
CountActors := RECORD
  FO.ActorName;
  Unsigned C := COUNT(GROUP)
END

MoviesIn := TABLE(FO, CountActors, ActorName);

OUTPUT (TOPN(MoviesIn, 100, -C));
```

LexisNexis® RISK SOLUTIONS

SQL vs. ECL (table and dataset)

SQL

```
CREATE TABLE Products (
  productCode VARCHAR(15) NOT NULL,
  productName VARCHAR(70) NOT NULL,
  productLine VARCHAR(50) NOT NULL,
  productScale VARCHAR(10) NOT NULL,
  productVendor VARCHAR(50) NOT NULL,
  productDescription TEXT NOT NULL,
  quantityInStock SMALLINT NOT NULL,
  buyPrice DOUBLE NOT NULL,
  MSRP DOUBLE NOT NULL,
  PRIMARY KEY (productCode)
);
```

ECL

```
Product := RECORD
    STRING productCode;
    STRING productName;
    STRING productLine;
    STRING productScale;
    STRING productVendor;
    STRING productDescription;
    INTEGER quantityInStock;
    DECIMAL7_2 buyPrice;
    DECIMAL7_2 MSRP;
END;

products := DATASET('~cm::products', Product , csv);
```

SQL vs. ECL (sort, count, group)

SQL

```
select * from products
 order by productName
```

```
select COUNT(*) from products
 where productLine='Vintage Cars'
```

```
select * from products
 group by productLine
```

ECL

```
SORT(products, productName);
```

```
COUNT(
 products(productLine='Vintage Cars'));
```

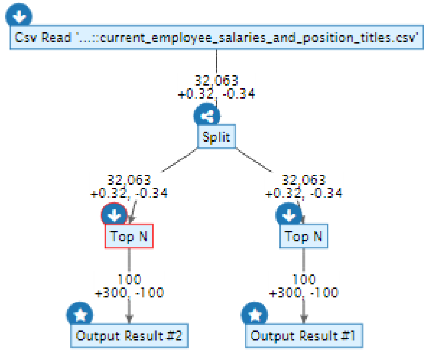
```
GROUP(
     SORT(products, productLine),
     productLine);
```

ECL: A Powerful Data Flow Language

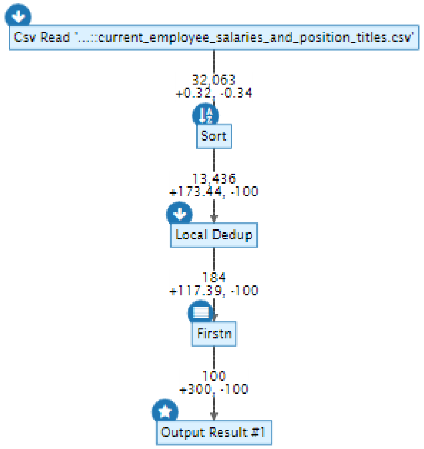
How you code

How the system executes it

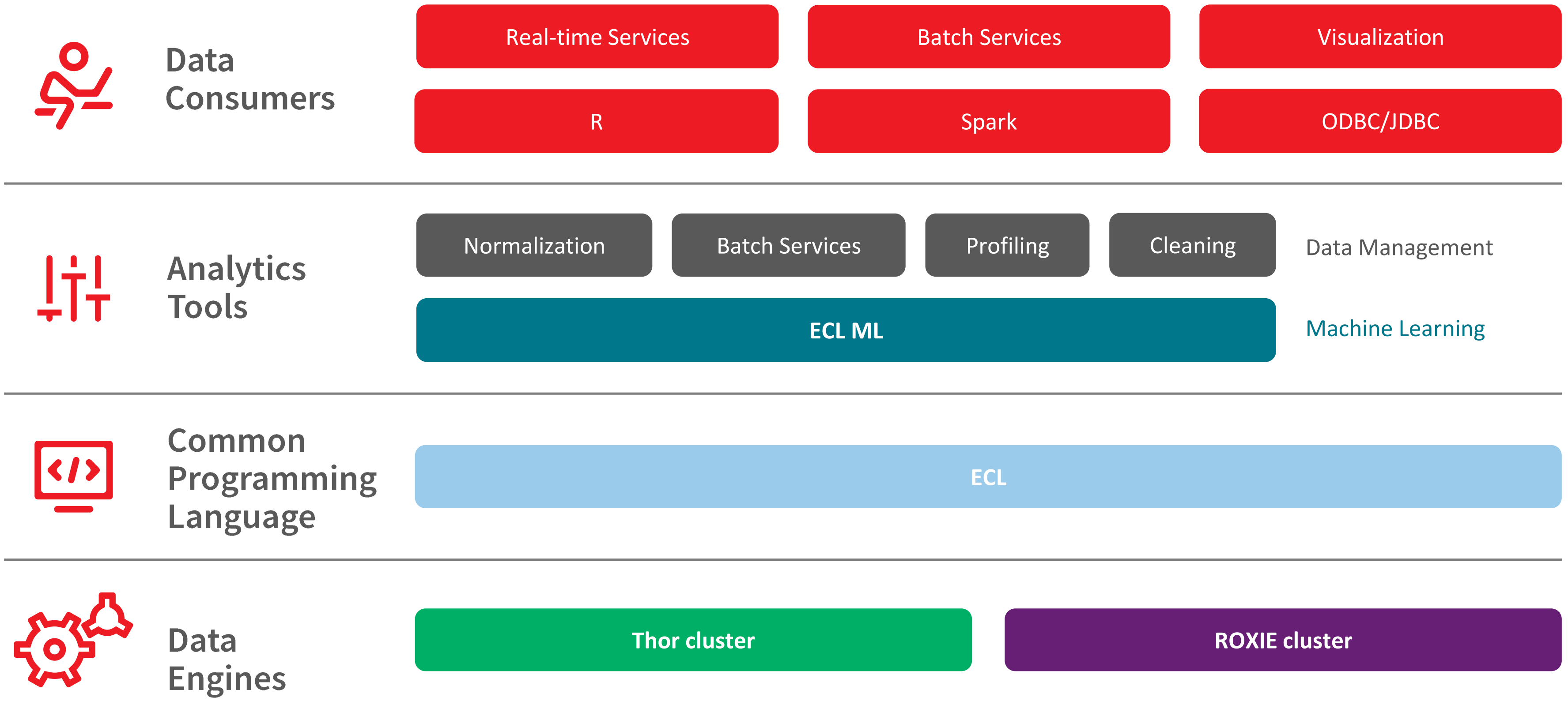
```
dsEmpSalary := DATASET(..);
OUTPUT(SORT(dsEmpSalary, position));
OUTPUT(SORT(dsEmpSalary, department));
```



```
dsEmpSalary := DATASET(..);
dsSortedEmp:=SORT(dsEmpSalary, position);
OUTPUT(DEDUP(dsSortedEmp, position));
```



Technology — **Layer View**



Enterprise Control Language (ECL)

ECL: What is it?

- Declarative Programming Language
  - "… a programming paradigm … that expresses the logic of a computation without describing its control flow." – *Wikipedia*
- Designed For Big Data Scenarios
- Any Cluster Size
- Source-to-source compiler
  - ECL code translated to C++ that is compiled to shared libraries and executed within a custom framework
- Can use external libraries and embedded code in certain other languages
  - C/C++, Python, R, Javascript
- Useful Documentation
  - Language Reference
    - https://hpccsystems.com/training/documentation/ecl-language-reference/html
  - Standard Library
    - https://hpccsystems.com/training/documentation/standard-library-reference/html

ECL: Basic, But Important, Stuff

- Two Statement Types
  - *Definition*
    - Assign an expression to an attribute
    - Example: `foo := bar * 2;`
  - *Action*
    - Do something that affects the outside world
      - Create a file, talk to a service, etc.
    - Example: `OUTPUT(foo, NAMED(‘foo_value’));`
- Plot Twist
  - You can define an attribute as an action
  - Example: `outputFoo := OUTPUT(foo, NAMED(‘foo_value’));`

ECL: Common Data Types

- Character
  - STRING[n]
  - UTF8
  - UNICODE[_locale][n]
- Numeric
  - INTEGER[n]
  - UNSIGNED[n]
  - REAL[n]
  - DECIMAL<n>[_y]
  - UDECIMAL<n>[_y]
- Other
  - BOOLEAN
  - SET OF <type>
  - RECORD
  - DATASET

ECL: Important Minutia

- Case-insensitive
- Whitespace insensitive (within reason)
- String literals are quoted with apostrophes
- Semicolon terminator
- C++/Java style commenting
- Definition (assignment) operator is `:=`
- Equality test operator is =
- Attributes can be defined only once
- Single-pass code parser
  - Only previously-defined attributes can be referenced
- Only those definitions that contribute to a result are actually used
- There are no loops

Stock Data

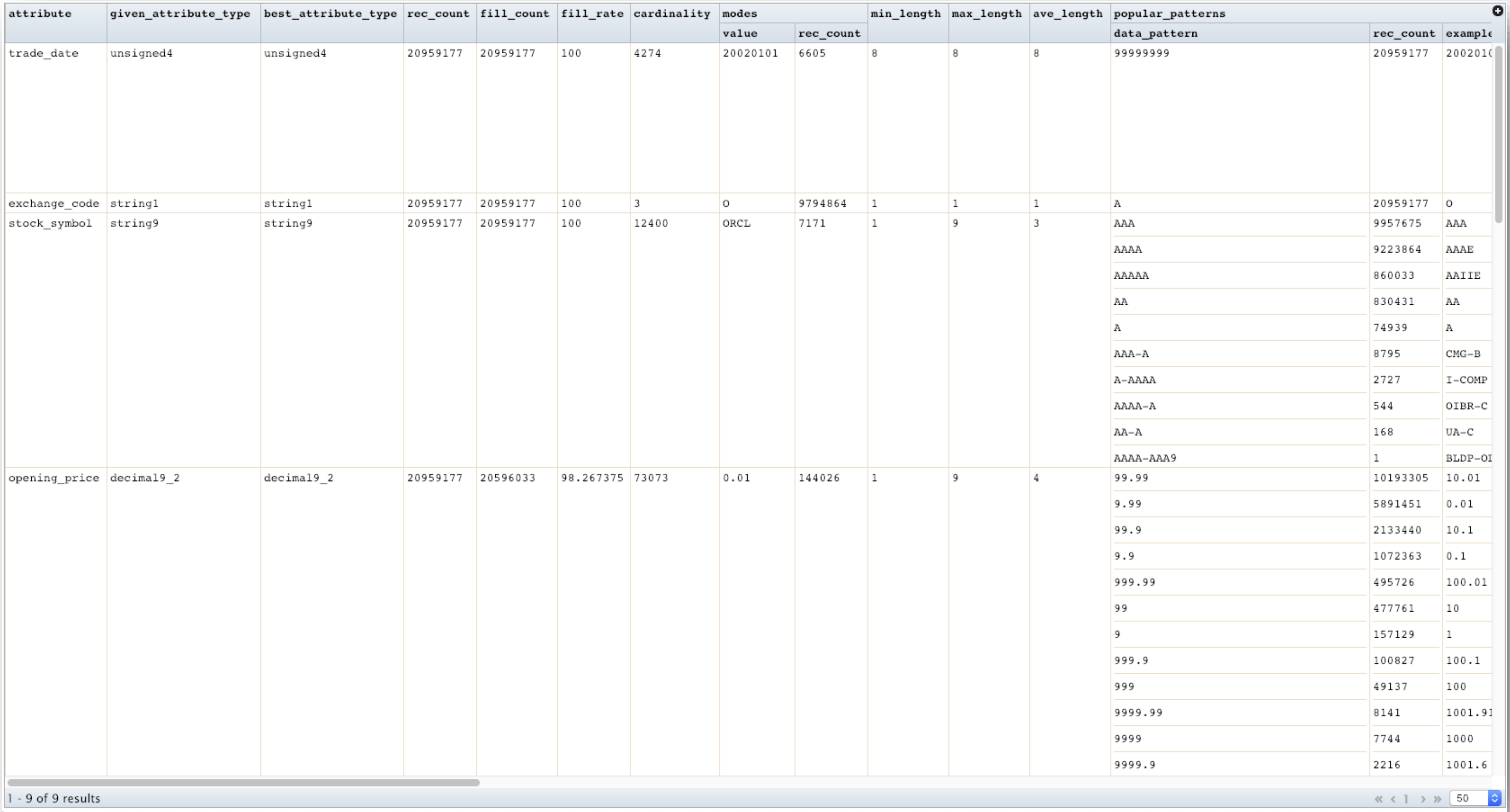
LexisNexis® RISK SOLUTIONS

Step 0: Raw Data

- 3 U.S. stock exchanges
  - NYSE
  - NASDAQ
  - AMEX
- 16 years of data
- 1GB of data in tab-delimited format
- ~21M rows of data

- Data fields:
  - trade_date
  - exchange_code
  - stock_symbol
  - opening_price
  - high_price
  - low_price
  - closing_price
  - shares_traded
  - share_value

Step 1: Profile the Data



attribute	given_attribute_type	best_attribute_type	rec_count	fill_count	fill_rate	cardinality	modes		min_length	max_length	ave_length	popular_patterns		
							value	rec_count				data_pattern	rec_count	example
trade_date	unsigned4	unsigned4	20959177	20959177	100	4274	20020101	6605	8	8	8	99999999	20959177	2002010
exchange_code	string1	string1	20959177	20959177	100	3	O	9794864	1	1	1	A	20959177	O
stock_symbol	string9	string9	20959177	20959177	100	12400	ORCL	7171	1	9	3	AAA	9957675	AAA
												AAAA	9223864	AAAE
												AAAAA	860033	AAIIE
												AA	830431	AA
												A	74939	A
												AAA-A	8795	CMG-B
												A-AAAA	2727	I-COMP
												AAAA-A	544	OIBR-C
												AA-A	168	UA-C
												AAAA-AAA9	1	BLDP-OI
opening_price	decimal9_2	decimal9_2	20959177	20596033	98.267375	73073	0.01	144026	1	9	4	99.99	10193305	10.01
												9.99	5891451	0.01
												99.9	2133440	10.1
												9.9	1072363	0.1
												999.99	495726	100.01
												99	477761	10
												9	157129	1
												999.9	100827	100.1
												999	49137	100
												9999.99	8141	1001.91
												9999	7744	1000
												9999.9	2216	1001.6

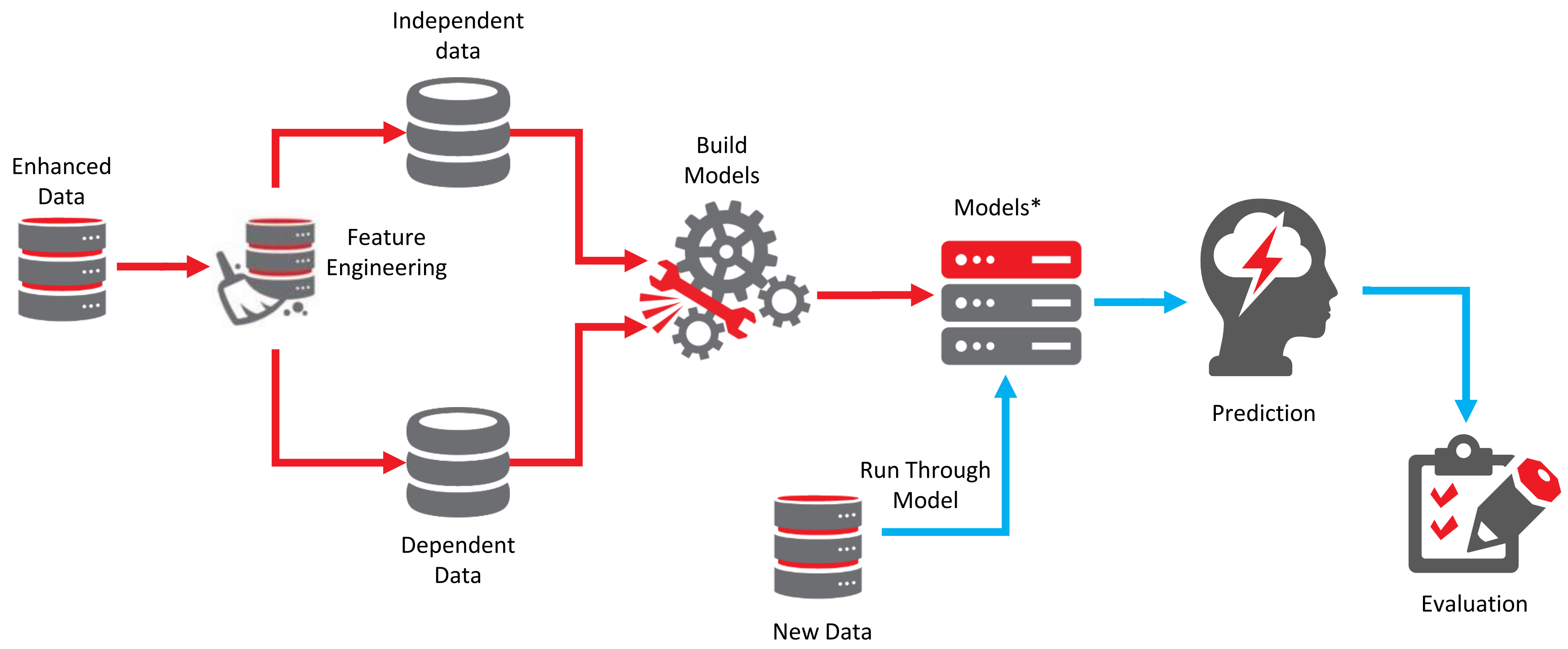
1 - 9 of 9 results

Step 2: Enhance the Data

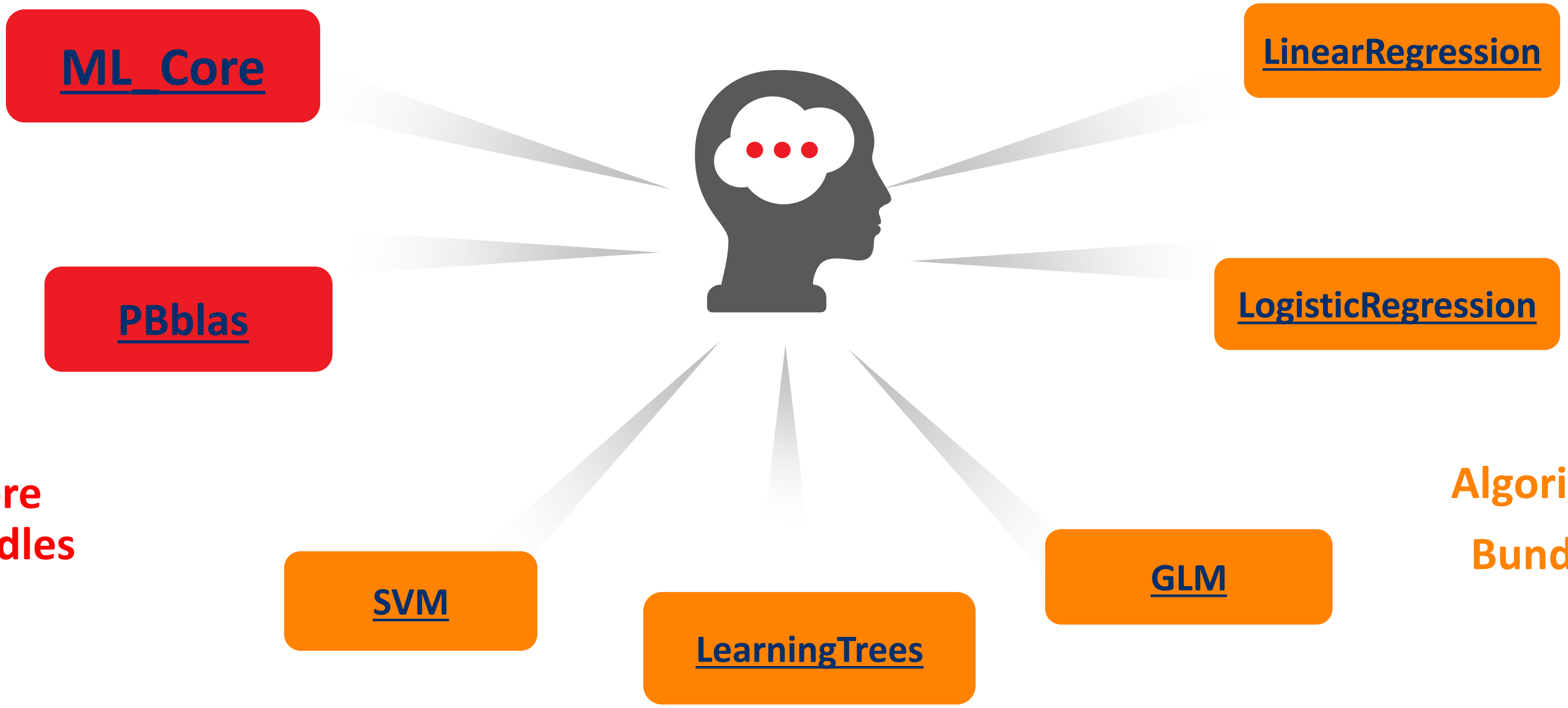
- Original Fields:
  - trade_date
  - exchange_code
  - stock_symbol
  - opening_price
  - high_price
  - low_price
  - closing_price
  - shares_traded
  - share_value

- Additional Fields:
  - symbol (exchange:stock_symbol)
  - trade_year
  - trade_month
  - trade_day
  - trade_day_of_week
  - trade_quarter
  - trade_day_of_year
  - trade_day_of_quarter
  - opening_price_change
  - closing_price_change
  - shares_traded_change
  - moving_ave_opening_price
  - moving_ave_high_price
  - moving_ave_low_price
  - moving_ave_closing_price
  - shares_traded_change_rate
  - direction

Step 3: Analytics



Step 3: Machine Learning on HPCC Systems Platform

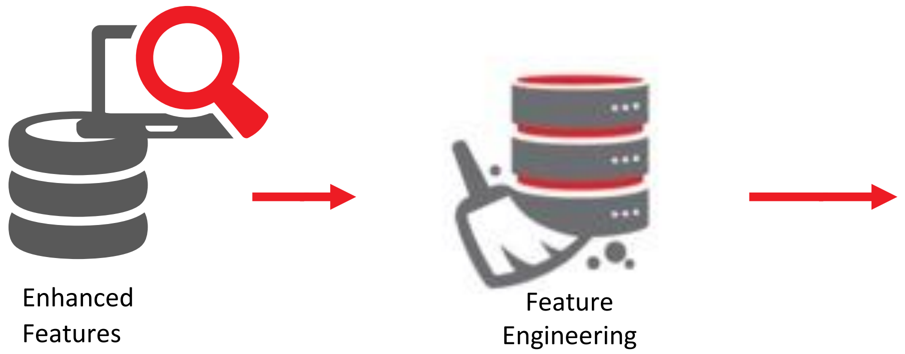


Core Bundles

Algorithm Bundles

LexisNexis® RISK SOLUTIONS

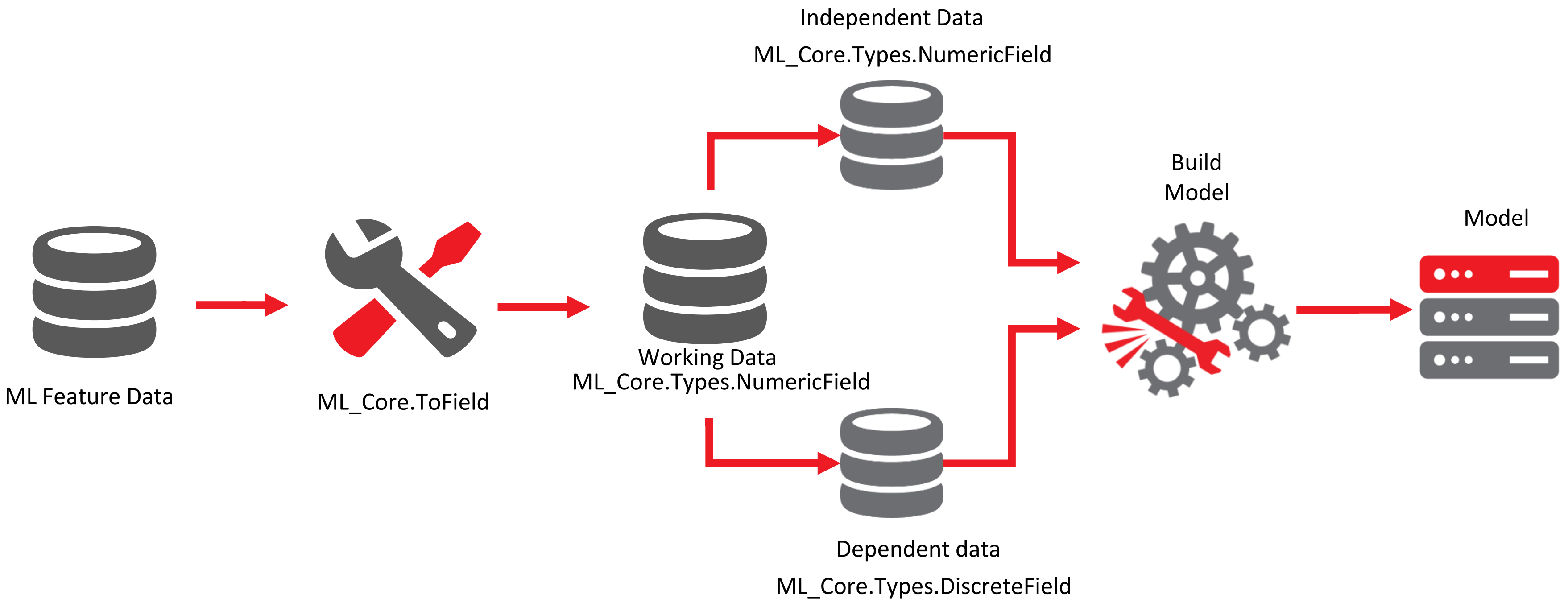
Step 3: Analytics Cont.,



- ML Feature Fields:
  - symbol (exchange:stock_symbol)
  - opening_price_change
  - closing_price_change
  - shares_traded_change
  - moving_ave_opening_price
  - moving_ave_high_price
  - moving_ave_low_price
  - moving_ave_closing_price
  - shares_traded_change_rate
  - direction

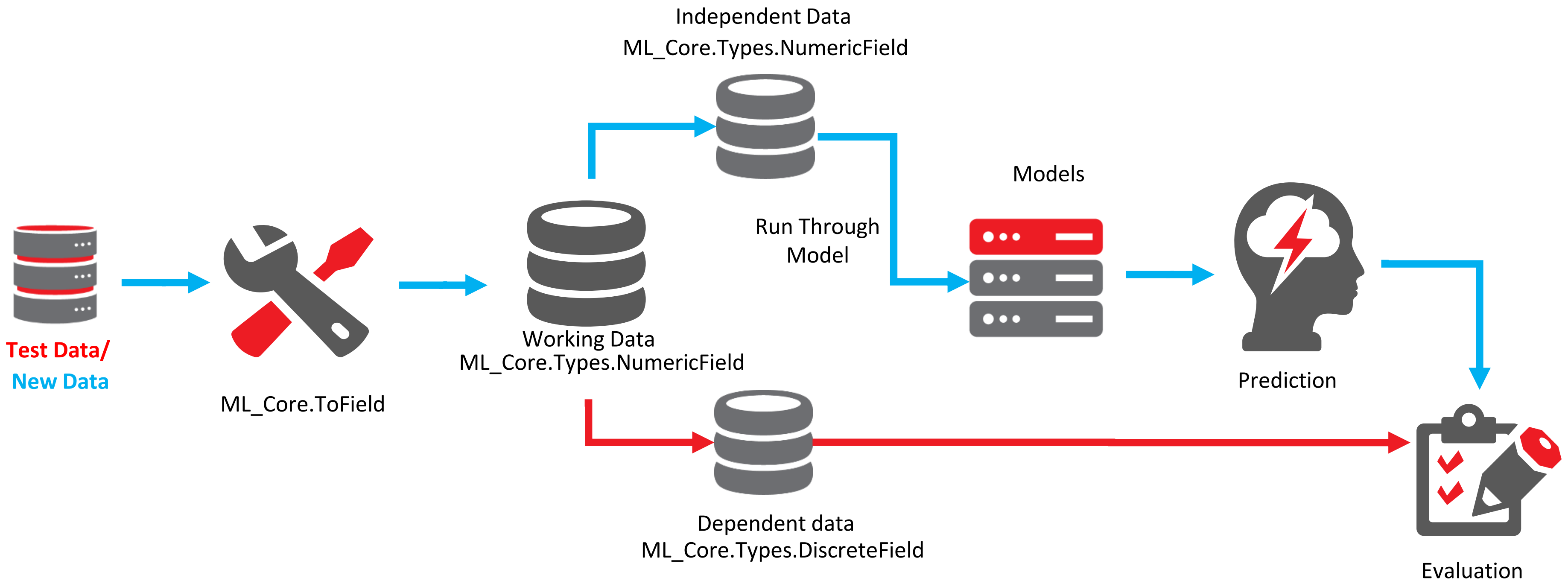
Direction	Description
0	Closing price is decreasing
1	Closing price is increasing

Step 3: Analytics Cont.,



LexisNexis® RISK SOLUTIONS

Step 3: Analytics Cont.,

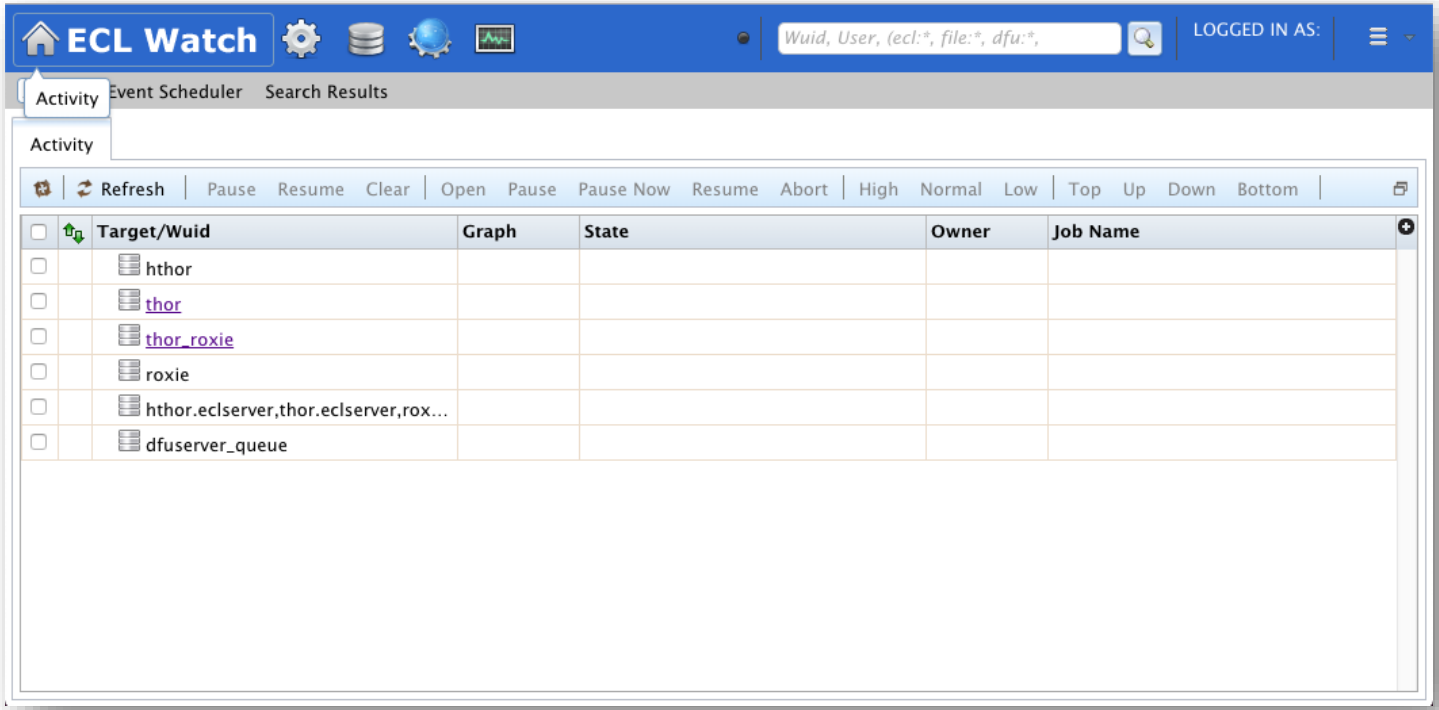


Setting Up Your Workstation

LexisNexis® RISK SOLUTIONS

Setting Up Your Workstation – Browser Bookmark

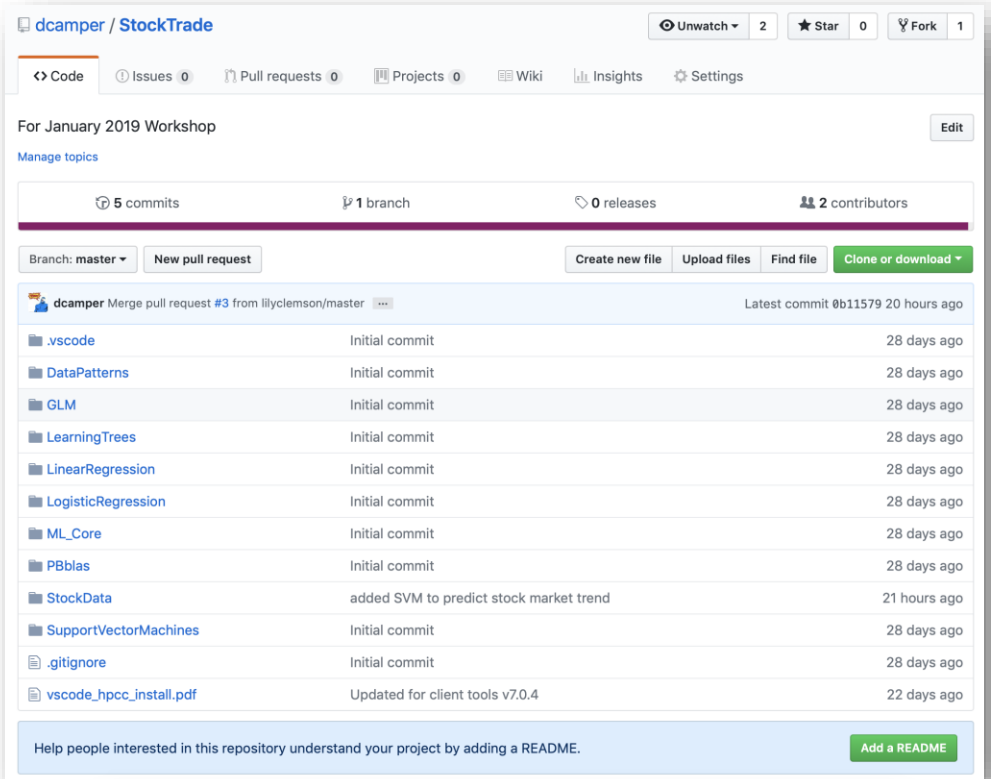
http://18.216.233.32:8010/



Target/Wuid	Graph	State	Owner	Job Name
hthor				
thor				
thor_roxie				
roxie				
hthor.eclserver,thor.eclserver,rox…				
dfuserver_queue				

Source Code and IDE Installation Instructions

https://github.com/dcamper/StockTrade



dcamper / StockTrade

For January 2019 Workshop

5 commits | 1 branch | 0 releases | 2 contributors

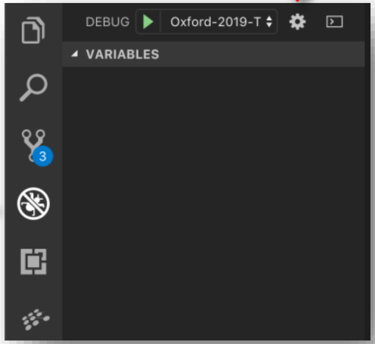
dcamper Merge pull request #3 from lilyclemson/master … Latest commit 0b11579 20 hours ago

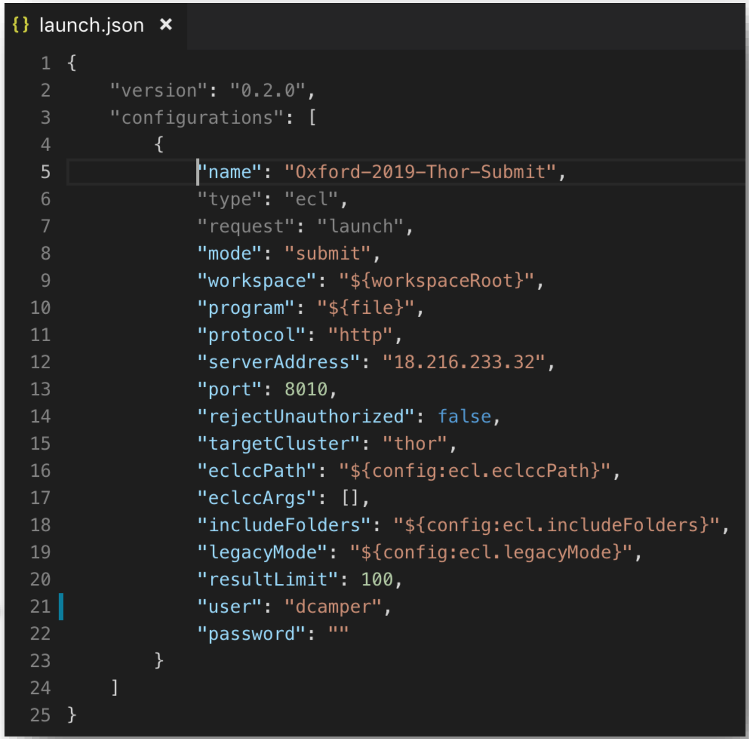
.vscode	Initial commit	28 days ago
DataPatterns	Initial commit	28 days ago
GLM	Initial commit	28 days ago
LearningTrees	Initial commit	28 days ago
LinearRegression	Initial commit	28 days ago
LogisticRegression	Initial commit	28 days ago
ML_Core	Initial commit	28 days ago
PBblas	Initial commit	28 days ago
StockData	added SVM to predict stock market trend	21 hours ago
SupportVectorMachines	Initial commit	28 days ago
.gitignore	Initial commit	28 days ago
vscode_hpcc_install.pdf	Updated for client tools v7.0.4	22 days ago

Help people interested in this repository understand your project by adding a README.

Source Code and IDE Installation Instructions

- Clone GitHub repo
  - https://github.com/dcamper/StockTrade
- Follow VS Code installation instructions
  - Do not install optional bundles
- Within VS Code, open the cloned StockTrade directory
- Modify the launch.json file
  - Change "user" value to your name
  - Save and close

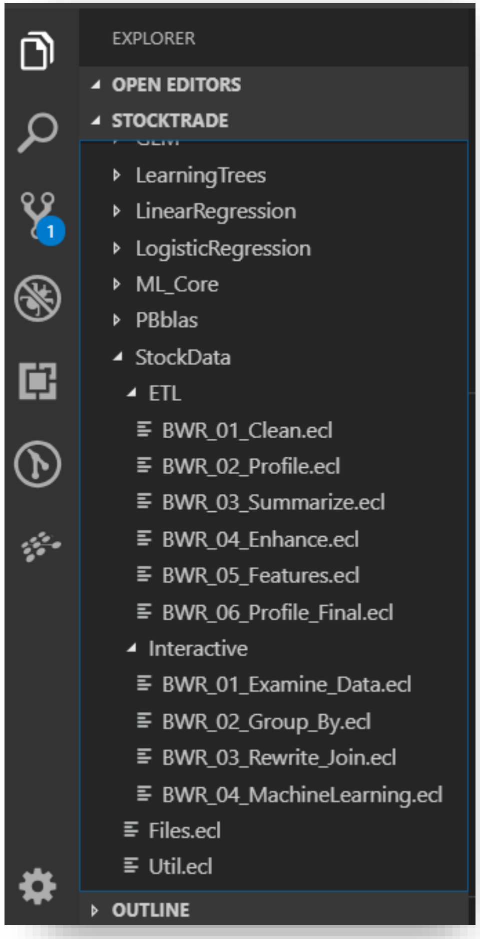




```
{
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Oxford-2019-Thor-Submit",
            "type": "ecl",
            "request": "launch",
            "mode": "submit",
            "workspace": "${workspaceRoot}",
            "program": "${file}",
            "protocol": "http",
            "serverAddress": "18.216.233.32",
            "port": 8010,
            "rejectUnauthorized": false,
            "targetCluster": "thor",
            "eclccPath": "${config:ecl.eclccPath}",
            "eclccArgs": [],
            "includeFolders": "${config:ecl.includeFolders}",
            "legacyMode": "${config:ecl.legacyMode}",
            "resultLimit": 100,
            "user": "dcamper",
            "password": ""
        }
    ]
}
```

Stock Data: Code

Code -> Purpose


EXPLORER
OPEN EDITORS
STOCKTRADE

- LearningTrees
- LinearRegression
- LogisticRegression
- ML_Core
- PBblas
- StockData
  - ETL
    - BWR_01_Clean.ecl
    - BWR_02_Profile.ecl
    - BWR_03_Summarize.ecl
    - BWR_04_Enhance.ecl
    - BWR_05_Features.ecl
    - BWR_06_Profile_Final.ecl
  - Interactive
    - BWR_01_Examine_Data.ecl
    - BWR_02_Group_By.ecl
    - BWR_03_Rewrite_Join.ecl
    - BWR_04_MachineLearning.ecl
  - Files.ecl
  - Util.ecl

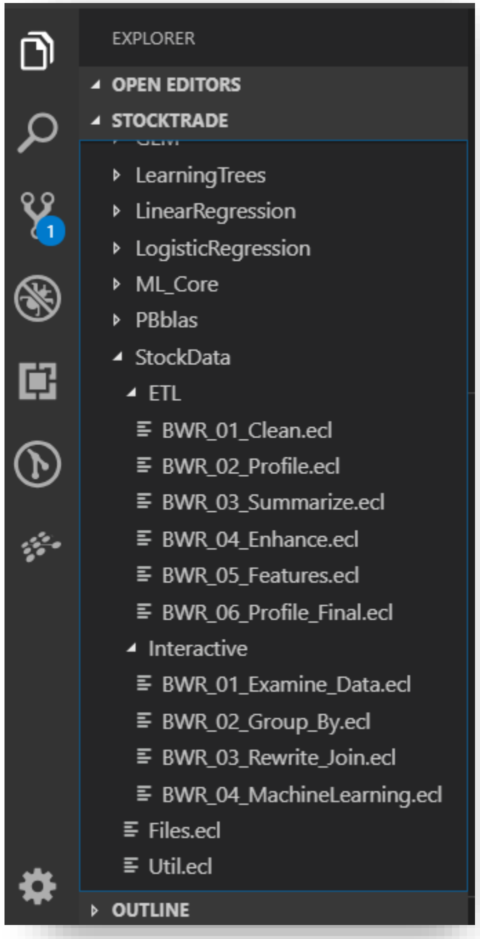
OUTLINE

Code used to clean, profile and enhance

Provided for review; please don't execute!

Example code you can run, modify, and play with

Code: ETL Directory Contents

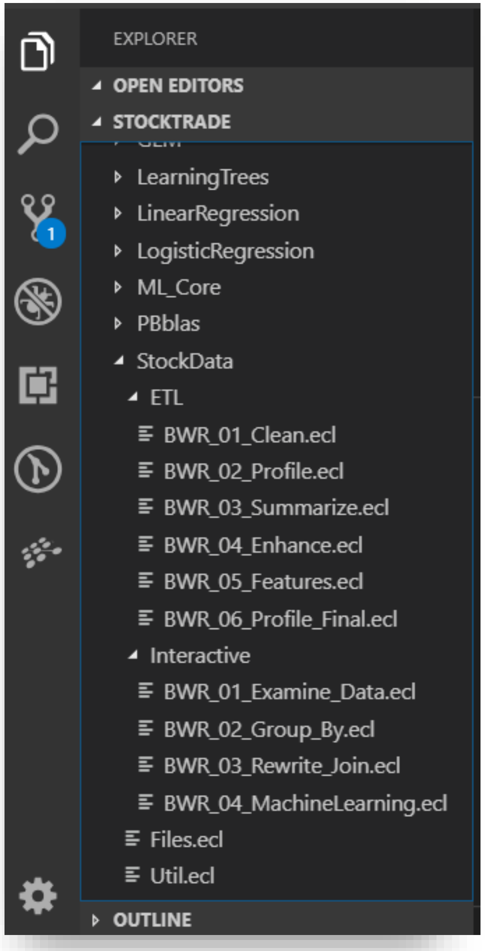


EXPLORER
- OPEN EDITORS
- STOCKTRADE
  - GLM
  - LearningTrees
  - LinearRegression
  - LogisticRegression
  - ML_Core
  - PBblas
  - StockData
    - ETL
      - BWR_01_Clean.ecl
      - BWR_02_Profile.ecl
      - BWR_03_Summarize.ecl
      - BWR_04_Enhance.ecl
      - BWR_05_Features.ecl
      - BWR_06_Profile_Final.ecl
    - Interactive
      - BWR_01_Examine_Data.ecl
      - BWR_02_Group_By.ecl
      - BWR_03_Rewrite_Join.ecl
      - BWR_04_MachineLearning.ecl
    - Files.ecl
    - Util.ecl
- OUTLINE

- BWR_01_Clean.ecl
  - *Simple rewrite of the raw data (where all fields were STRING) as a fully datatyped dataset*
- BWR_02_Profile.ecl
  - *Perform data profiling against the original data that has has datatyped fields*
- BWR_03_Summarize.ecl
  - *Create some per-symbol statistics and write the results to a separate file*
- BWR_04_Enhance.ecl
  - *Pull apart the data in certain fields, like date, and create new fields containing features*
- BWR_05_Features.ecl
  - *Append some other features that will be used for machine learning*
- BWR_06_Profile_File.ecl
  - *Perform data profiling against the final dataset, which includes appended fields*

LexisNexis® RISK SOLUTIONS

Code: Interactive Directory Contents



```
EXPLORER
◢ OPEN EDITORS
◢ STOCKTRADE
  ▷ LearningTrees
  ▷ LinearRegression
  ▷ LogisticRegression
  ▷ ML_Core
  ▷ PBblas
  ◢ StockData
    ◢ ETL
       BWR_01_Clean.ecl
       BWR_02_Profile.ecl
       BWR_03_Summarize.ecl
       BWR_04_Enhance.ecl
       BWR_05_Features.ecl
       BWR_06_Profile_Final.ecl
    ◢ Interactive
       BWR_01_Examine_Data.ecl
       BWR_02_Group_By.ecl
       BWR_03_Rewrite_Join.ecl
       BWR_04_MachineLearning.ecl
     Files.ecl
     Util.ecl
▷ OUTLINE
```

- BWR_01_Examine_Data.ecl
  - *Output a sample of the data and some minor statistics about it*
- BWR_02_Group_By.ecl
  - *Demonstrate the TABLE() function by computing the average number of shares traded for each day of the week*
- BWR_03_Rewrite_Join.ecl
  - *Append a new field that shows how the closing price compares against that year's median value*
- BWR_04_MachineLearning.ecl
  - *Perform a logistic regression against the movement of Apple's closing price*

Let’s Play With The Code

LexisNexis® RISK SOLUTIONS

Useful Links

- Open Source HPCC Systems Platform: Home Page
- Internship Program
- Online Training
- Download Page
- Our GitHub portal
- Community Forums
- Getting Started with ECL
- Advanced ECL
- Latest Release and Documentation
- Supported plugins, connectors, third party modules and bundles
- Machine Learning on HPCC Systems

Join our Community

Help us make HPCC Systems better. Register on our community portal.

Questions?

LexisNexis®

RISK SOLUTIONS