Machine Learning

from Development to Production at Instacart

Montana Low
Machine Learning Engineer, Instacart

Instacart value proposition

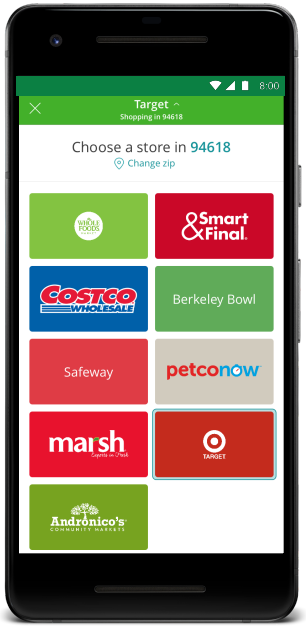
Groceries + From stores you love + Delivered to your doorstep + In as little as an hour =

instacart

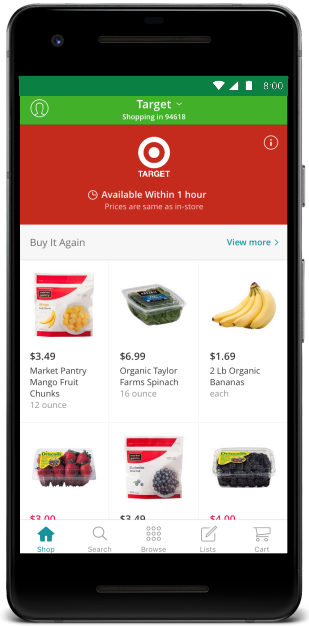
Four sided marketplace

instacart

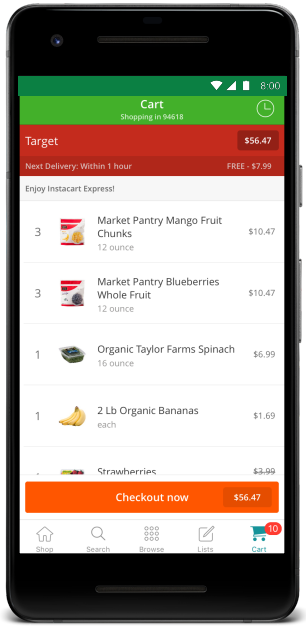
Customer experience



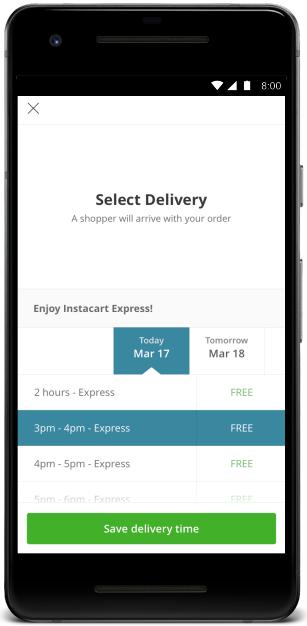
Choose a store



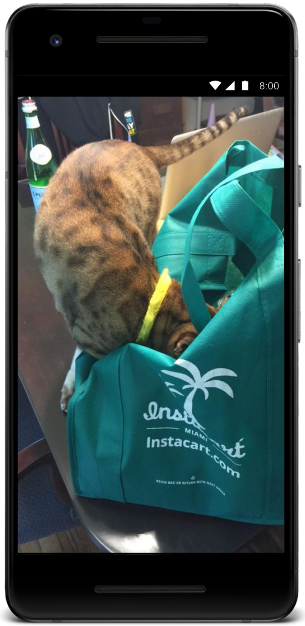
Shop for groceries



Checkout



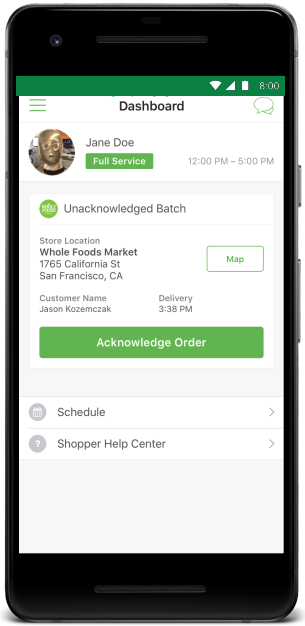
Select delivery time



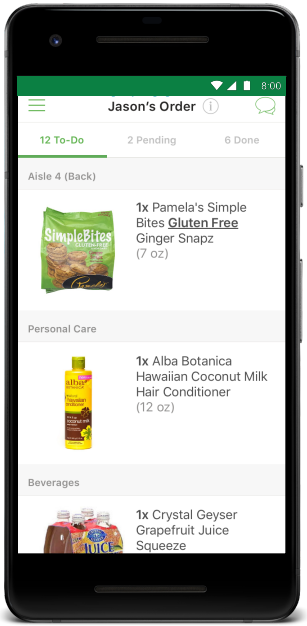
Delivered to doorstep

instacart

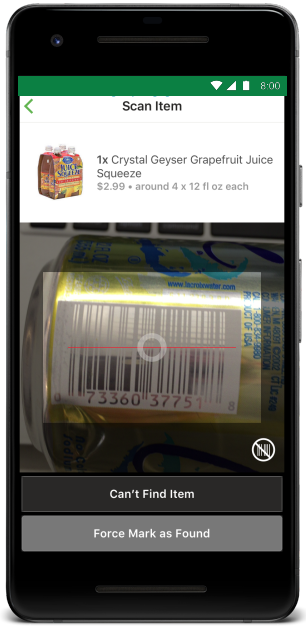
Personal shopper experience



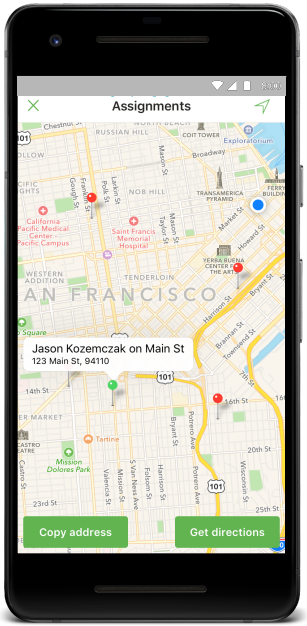
Accept order



Find the groceries



Scan barcode



Out for delivery

Delivered to doorstep

instacart

Search & discovery

Supervised learning

Milk

Milk chocolate

Chocolate milk

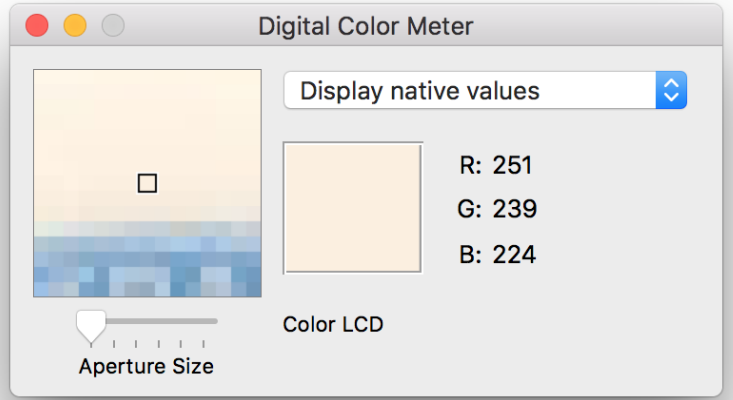
instacart

Features

- Brand
- Fat Content
- USDA Grade
- Organic?
- Pasteurized?
- Homogenized?
- Volume
- Geography
- …
- Dominant Color

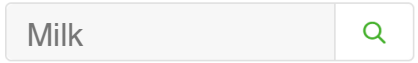
instacart

Encoding



instacart

Supervised learning



instacart

New products

- Kirkland signature
- 2% Fat
- Milk
- Vitamin A
- Vitamin D
- Grade A
- Pasteurized
- Homogenized
- 1 gallon
- 2 count
- …
- Tertiary color

instacart

Competitive products

Cola

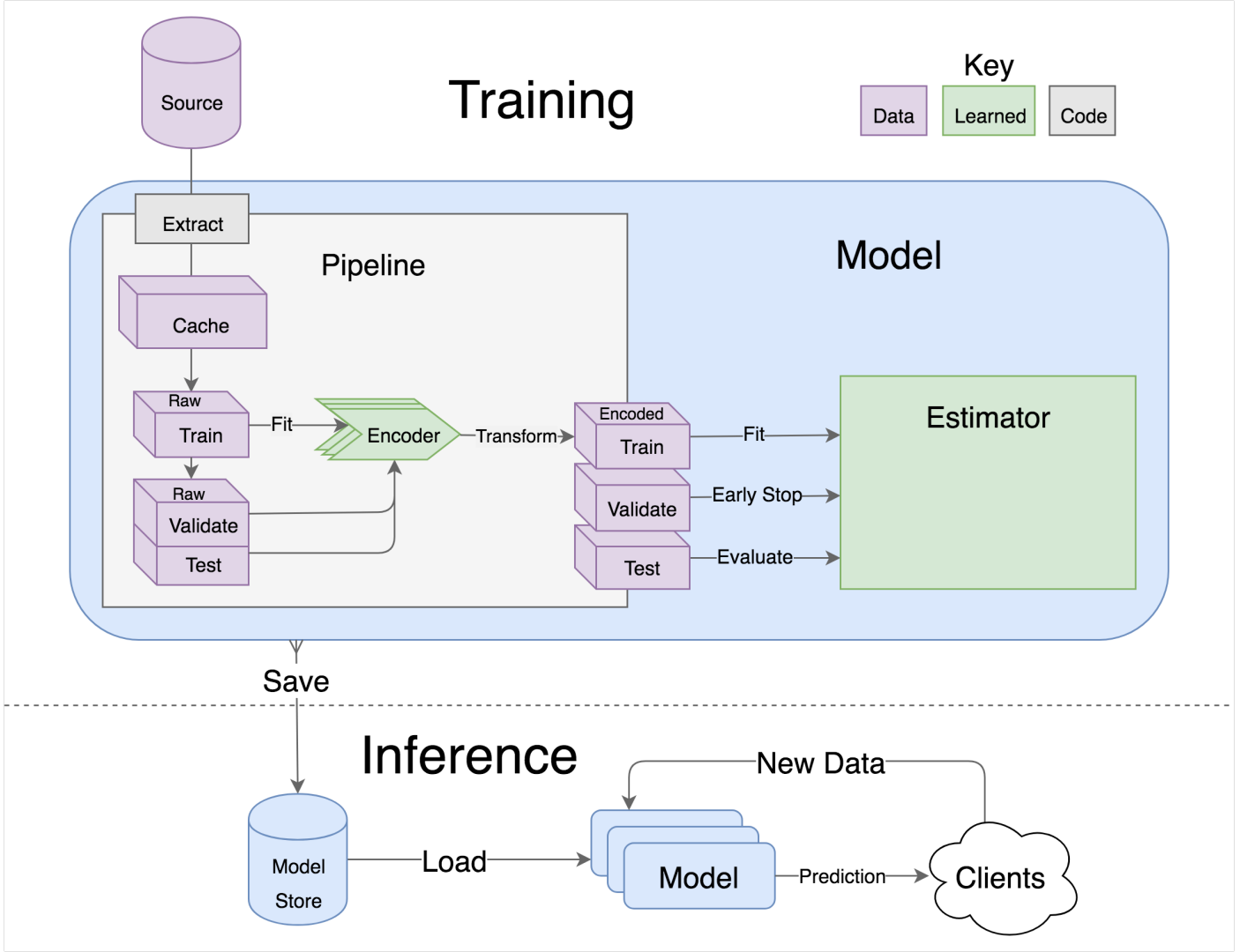
instacart

Recommended products

Peanut butter

instacart

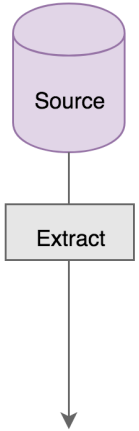
Project implementation w/ Lore


Training
Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Raw
Validate
Early Stop
Validate
Test
Evaluate
Test
Save
Inference
New Data
Model
Store
Load
Model
Prediction
Clients

instacart

Create a project and model

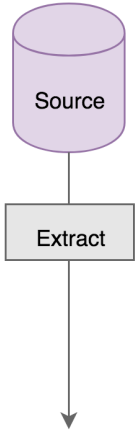
```
$ pip install lore
$ lore init loss_prevention
$ lore generate scaffold delivery_disputes --regression

loss_prevention in development on montanalow@localhost
CREATED loss_prevention/models/delivery_disputes.py
CREATED loss_prevention/estimators/delivery_disputes.py
CREATED loss_prevention/pipelines/delivery_disputes.py
CREATED tests/unit/test_delivery_disputes.py
CREATED notebooks/delivery_disputes/features.ipynb
CREATED notebooks/delivery_disputes/architecture.ipynb
```

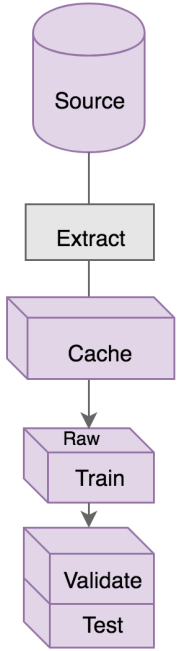


Extract

loss_prevention/extracts/credit_card_disputes.sql

```
SELECT
  visits.ip_address,
  deliveries.latitude,
  deliveries.longitude,
  charge_logs.is_disputed
FROM deliveries
JOIN visits ON visits.id = deliveries.visit_id
JOIN charge_logs ON charge_logs.id = deliveries.charge_id
```




Source
Extract
Cache
Raw
Train
Validate
Test

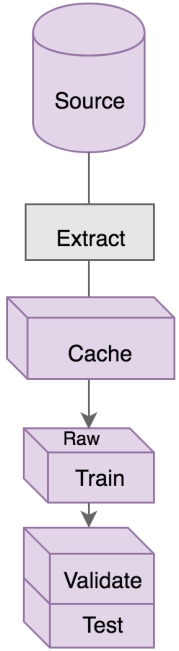

Key
Data
Learned
Code

instacart

Pipeline

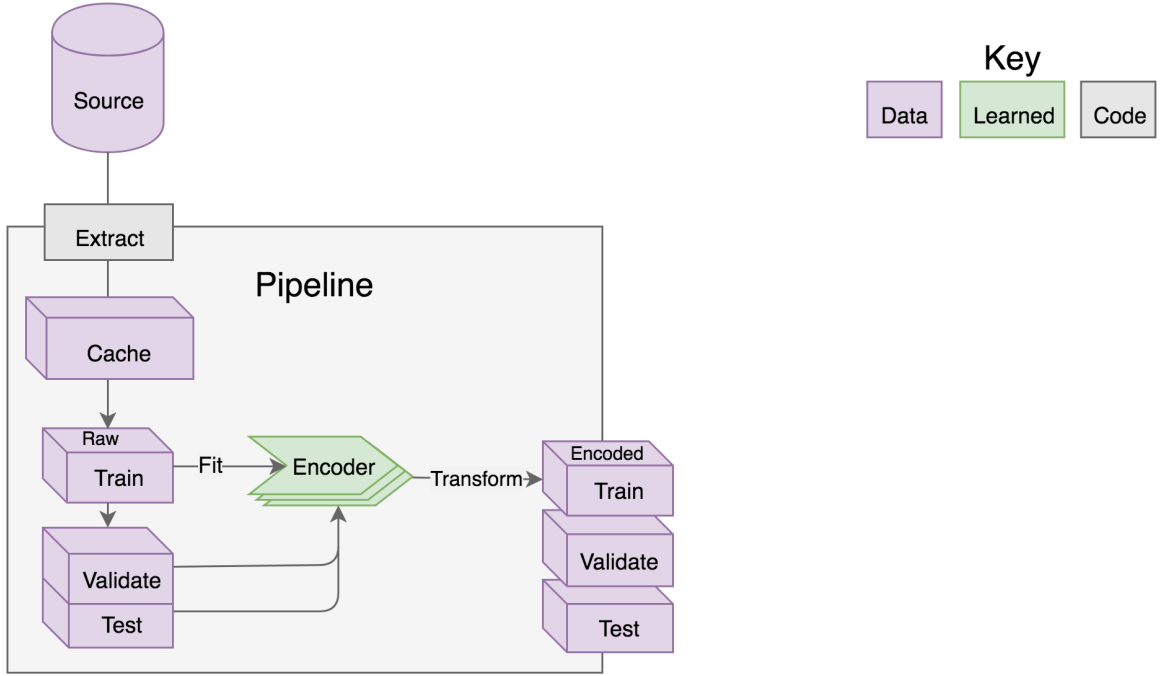
loss_prevention/pipelines/credit_card_disputes.py

```
class Pipeline(lore.pipelines.holdout.Base):

    @timed(logging.INFO)
    def get_data(self):
        return lore.io.redshift.dataframe(
            filename='credit_card_disputes',
            cache=True
        )
```


Source
Extract
Cache
Raw
Train
Validate
Test


Key
Data
Learned
Code

instacart


Source
Extract
Pipeline
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Validate
Test
Validate
Test
Key
Data
Learned
Code

Pipeline

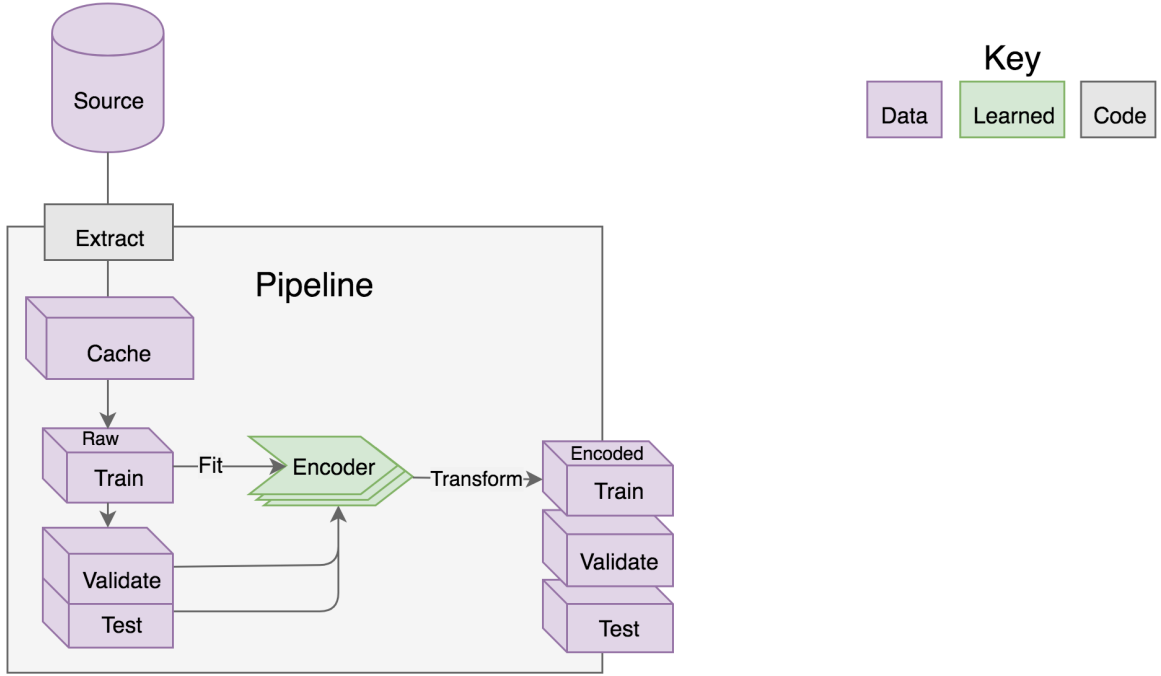
loss_prevention/pipelines/credit_card_disputes.py

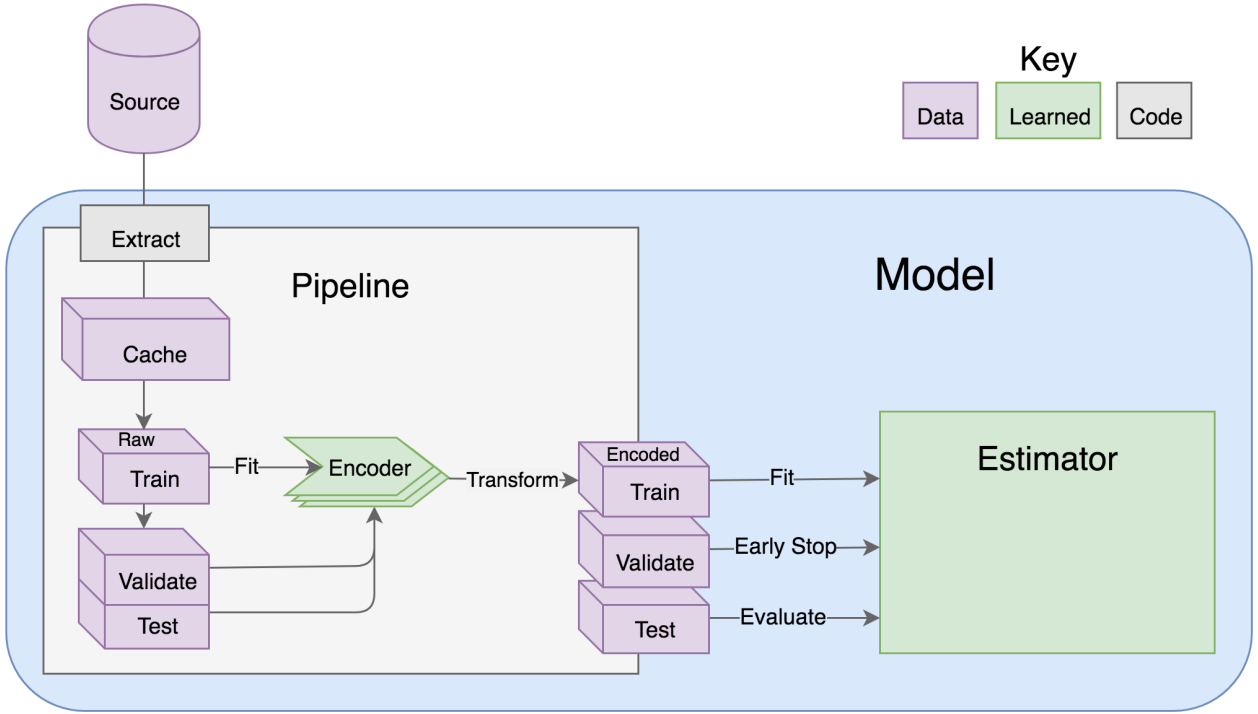
```
class Pipeline(lore.pipelines.holdout.Base):
    ...

    def get_encoders(self):
        return (
            Norm(
                    Distance(
                        'latitude',
                        'longitude',
                        GeoIP('ip_address', 'latitude'),
                        GeoIP('ip_address', 'longitude')
                    )
            ),
        )
```

Pipeline

loss_prevention/pipelines/credit_card_disputes.py

```
class Pipeline(lore.pipelines.holdout.Base):
    ...

    def get_output_encoder(self):
        return Pass('is_disputed')
```

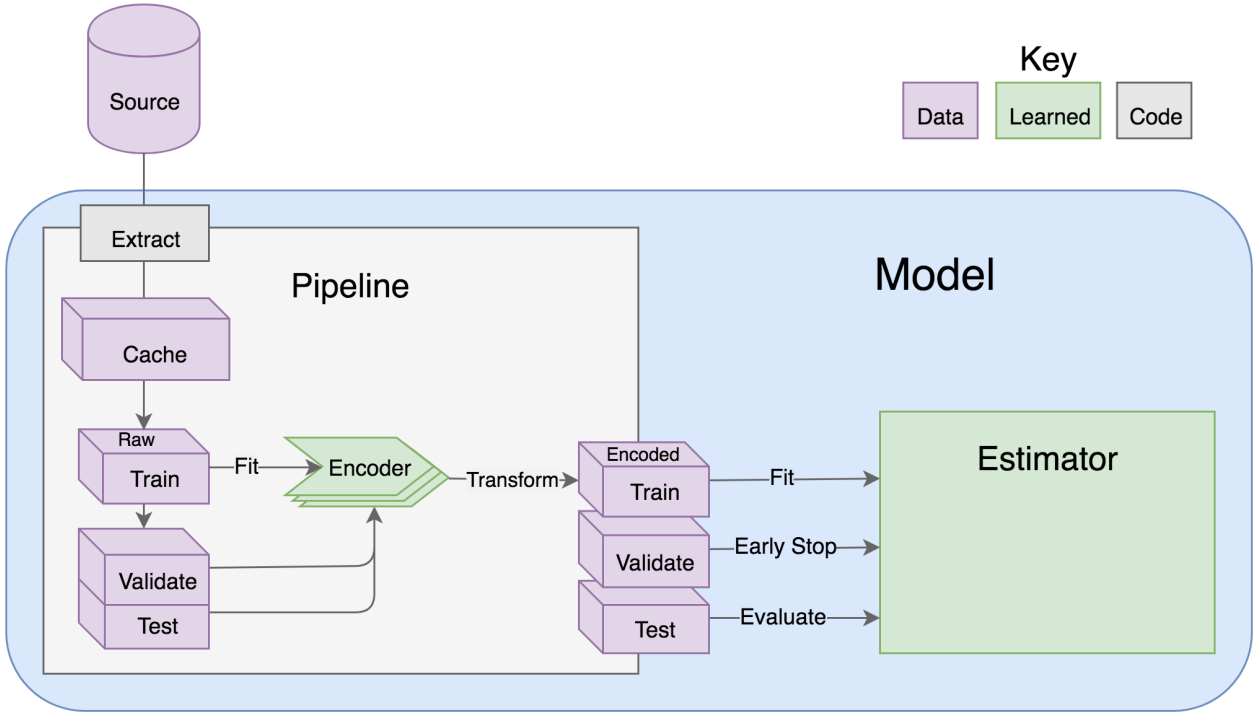

Source
Extract
Pipeline
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Validate
Test
Validate
Test
Key
Data
Learned
Code


Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Validate
Early Stop
Validate
Test
Evaluate
Test

Model

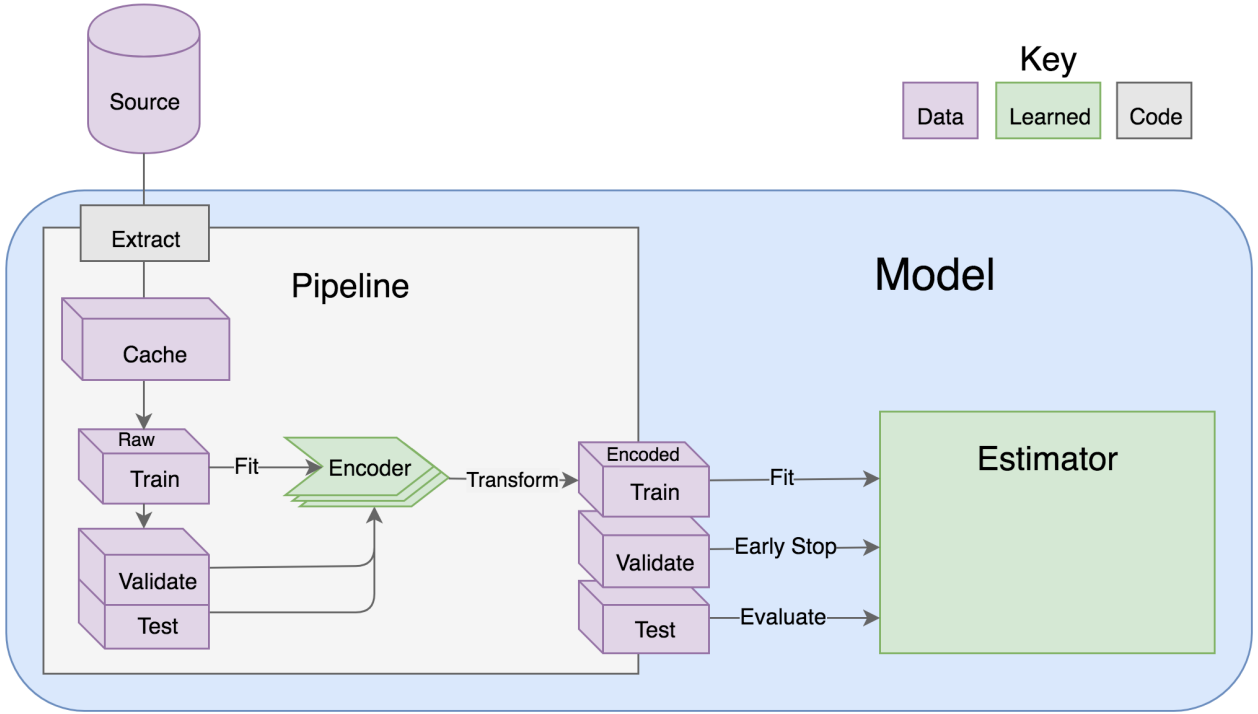
loss_prevention/models/credit_card_disputes.py

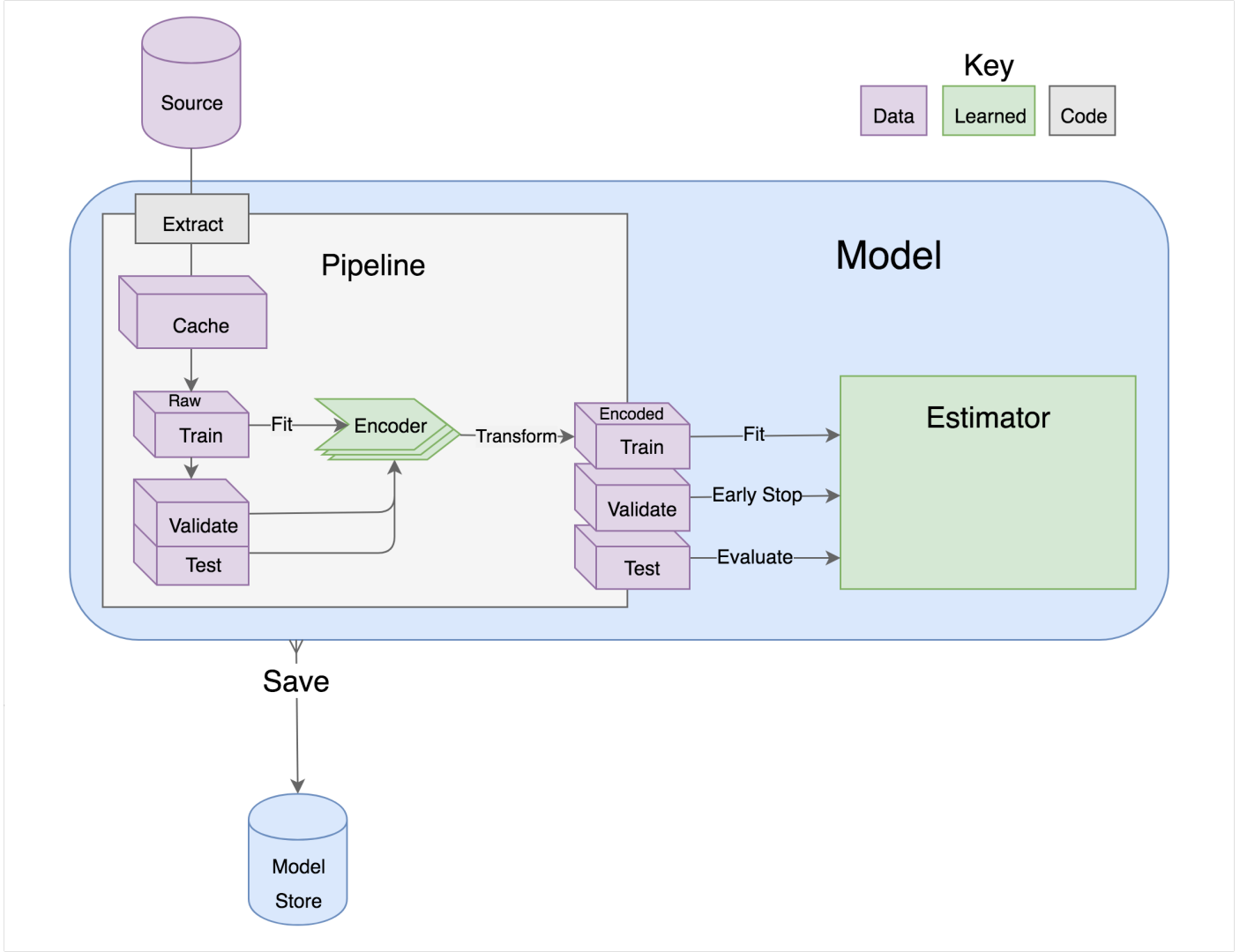
```
from loss_prevention.pipelines.credit_card_diputes import Pipeline

class DeepLearning(lore.models.keras.Base):
  def __init__(self):
    super(DeepLearning, self).__init__(
      pipeline=Pipeline(),
      estimator=lore.estimators.keras.BinaryClassifier()
    )
```


Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Validate
Early Stop
Validate
Test
Evaluate
Test

Run tests

```
$ lore test

loss_prevention in test on montanalow@localhost
RUNNING all tests
..
----------------------------------------------
Ran 2 tests in 3.846s

OK
```


Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Validate
Early Stop
Validate
Test
Evaluate
Test


Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Validate
Early Stop
Validate
Test
Evaluate
Test
Save
Model
Store

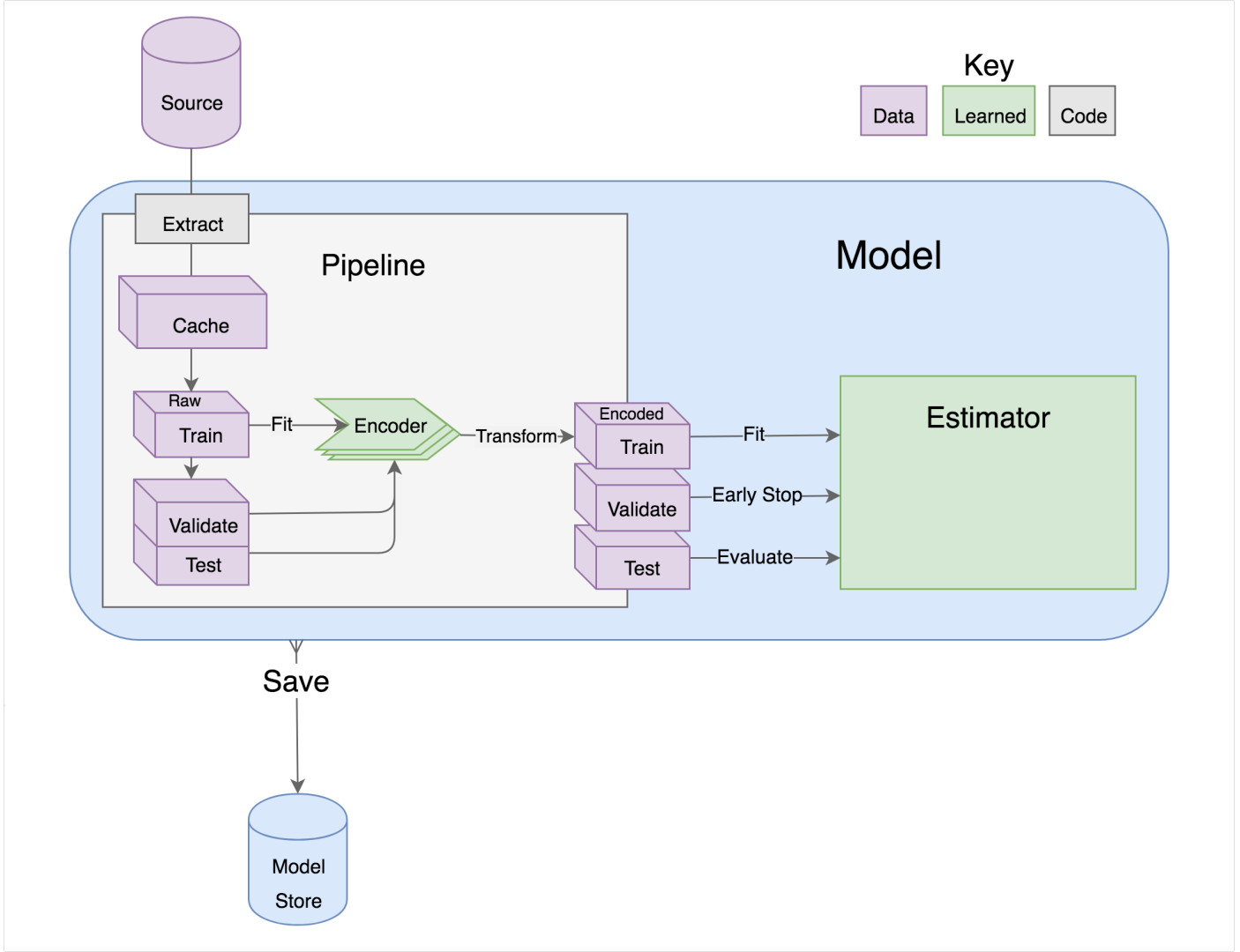
Train the model

```
$ lore fit loss_prevention.models.delivery_disputes.DeepLearning

loss_prevention in development on montanalow@localhost
Using TensorFlow backend. Train on 80 samples, validate on 10 samples

Epoch 1 32/80 [===========>.............] - ETA: 15s - loss: 1.5831
```

instacart

Early Stopping

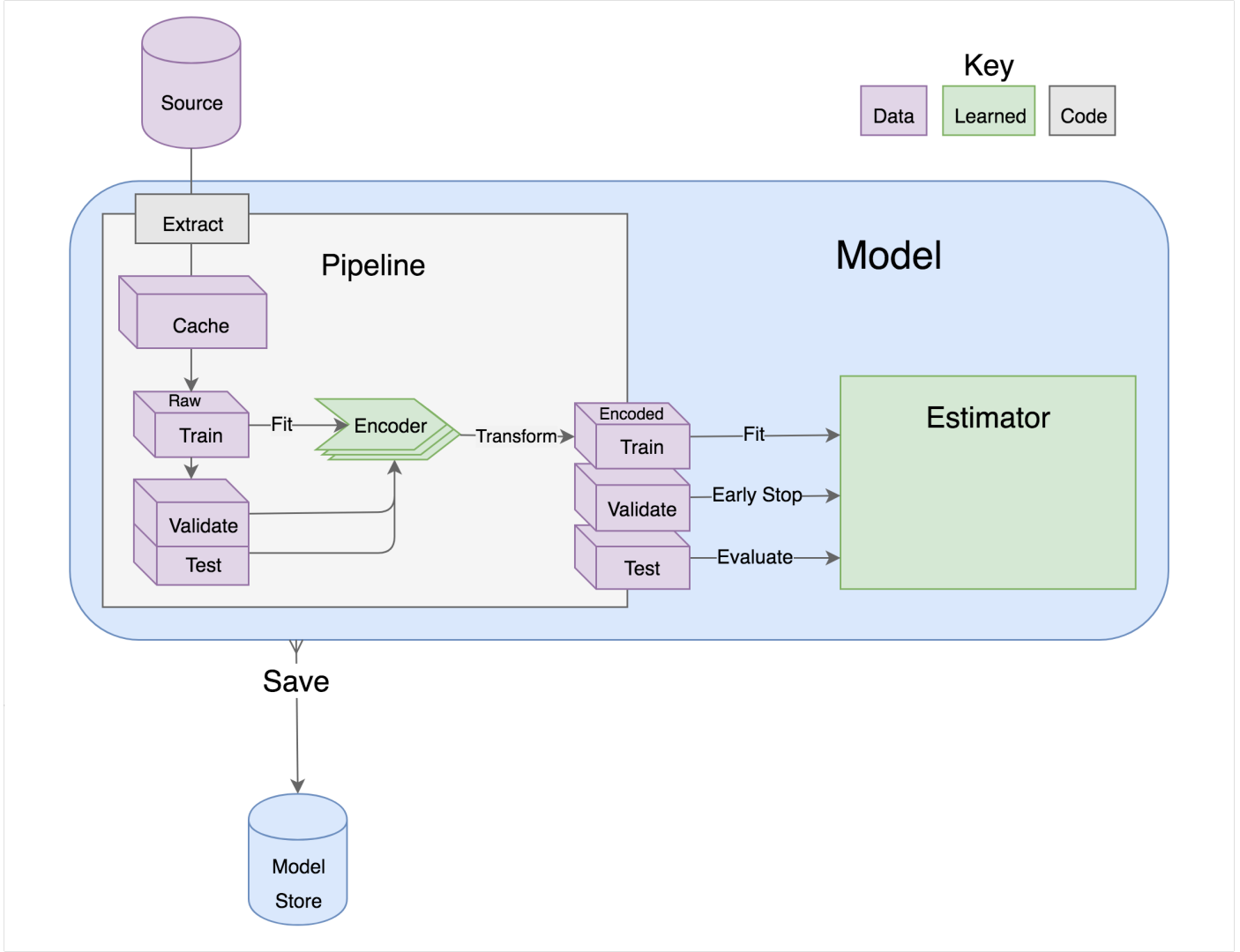
```
$ lore fit loss_prevention.models.delivery_disputes.DeepLearning

loss_prevention in development on montanalow@localhost
Using TensorFlow backend. Train on 80 samples, validate on 10 samples

...

Epoch 57 80/80 [==========================] - loss: 0.55 val_loss: 0.58

Epoch 58 80/80 [==========================] - loss: 0.53 val_loss: 0.57

Epoch 59 80/80 [==========================] - loss: 0.52 val_loss: 0.58

Early Stopping
```

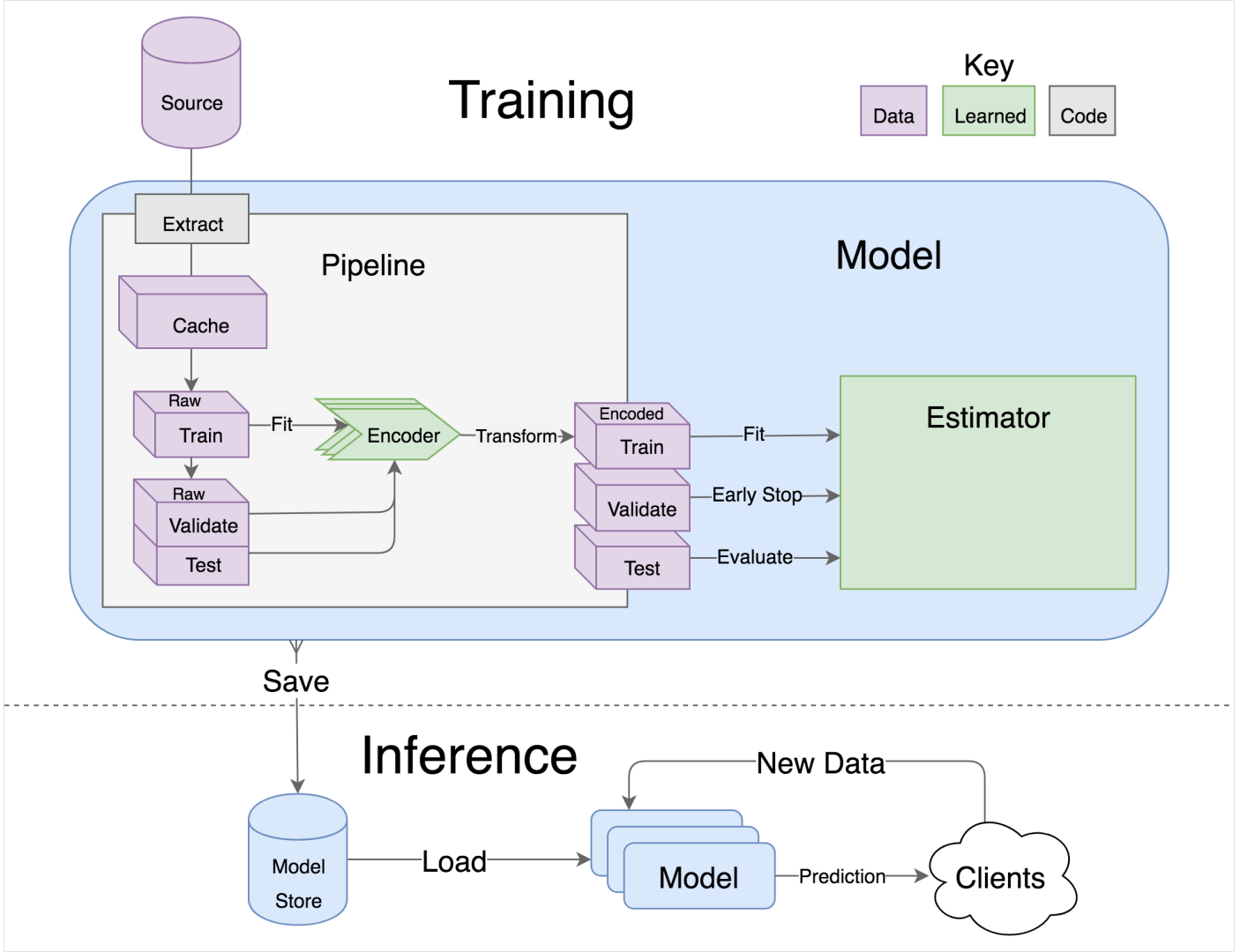

Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Validate
Early Stop
Validate
Test
Evaluate
Test
Save
Model
Store

Important Files

```
requirements.txt
runtime.txt

config/
  database.cfg

data/query_cache/
  loss_prevention.pipelines.delivery_disputes.Pipeline.get_data.XY.pickle

models/loss_prevention.models.delivery_disputes/DeepLearning/1/
  model.pickle
  weights.h5

logs/
  development.log
```

instacart


Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Validate
Early Stop
Validate
Test
Evaluate
Test
Save
Model
Store


Training
Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Raw
Validate
Early Stop
Validate
Test
Evaluate
Test
Save
Inference
New Data
Model
Store
Load
Model
Prediction
Clients

instacart

Extract

loss_prevention/extracts/credit_card_disputes.sql

```
SELECT
  visits.ip_address,
  deliveries.latitude,
  deliveries.longitude,
  charge_logs.is_disputed
FROM deliveries
JOIN visits ON visits.id = deliveries.visit_id
JOIN charge_logs ON charge_logs.id = deliveries.charge_id
```

Extract

loss_prevention/extracts/credit_card_disputes.sql.j2

```
SELECT
  visits.ip_address,
  deliveries.latitude,
  deliveries.longitude,
  charge_logs.is_disputed
FROM deliveries
JOIN visits ON visits.id = deliveries.visit_id
JOIN charge_logs ON charge_logs.id = deliveries.charge_id
{% if delivery_id %}
WHERE deliveries.id = {delivery_id}
{% endif %}
```

Pipeline

loss_prevention/pipelines/credit_card_disputes.py

```
class Pipeline(lore.pipelines.holdout.Base):

    @timed(logging.INFO)
    def get_data(self):
        return lore.io.redshift.dataframe(
            filename='credit_card_disputes',
            cache=True
        )
```

Pipeline

loss_prevention/pipelines/credit_card_disputes.py

```
class Pipeline(lore.pipelines.holdout.Base):

    @timed(logging.INFO)
    def get_data(self, delivery_id=None):
        if delivery_id:
            interpolate = {'delivery_id': delivery_id}
            connection = lore.io.postgres
            cache = False
        else:
            interpolate = {}
            connection = lore.io.redshift
            cache = True

        sql=connection.template('delivery_disputes', delivery_id=delivery_id)
        return connection.dataframe(sql=sql, cache=cache, **interpolate)
```

instacart

Model

loss_prevention/models/credit_card_disputes.py

```
from loss_prevention.pipelines.credit_card_diputes import Pipeline

class DeepLearning(lore.models.keras.Base):
  def __init__(self):
    super(DeepLearning, self).__init__(
      pipeline=Pipeline(),
      estimator=lore.estimators.keras.BinaryClassifier()
    )
```

Model

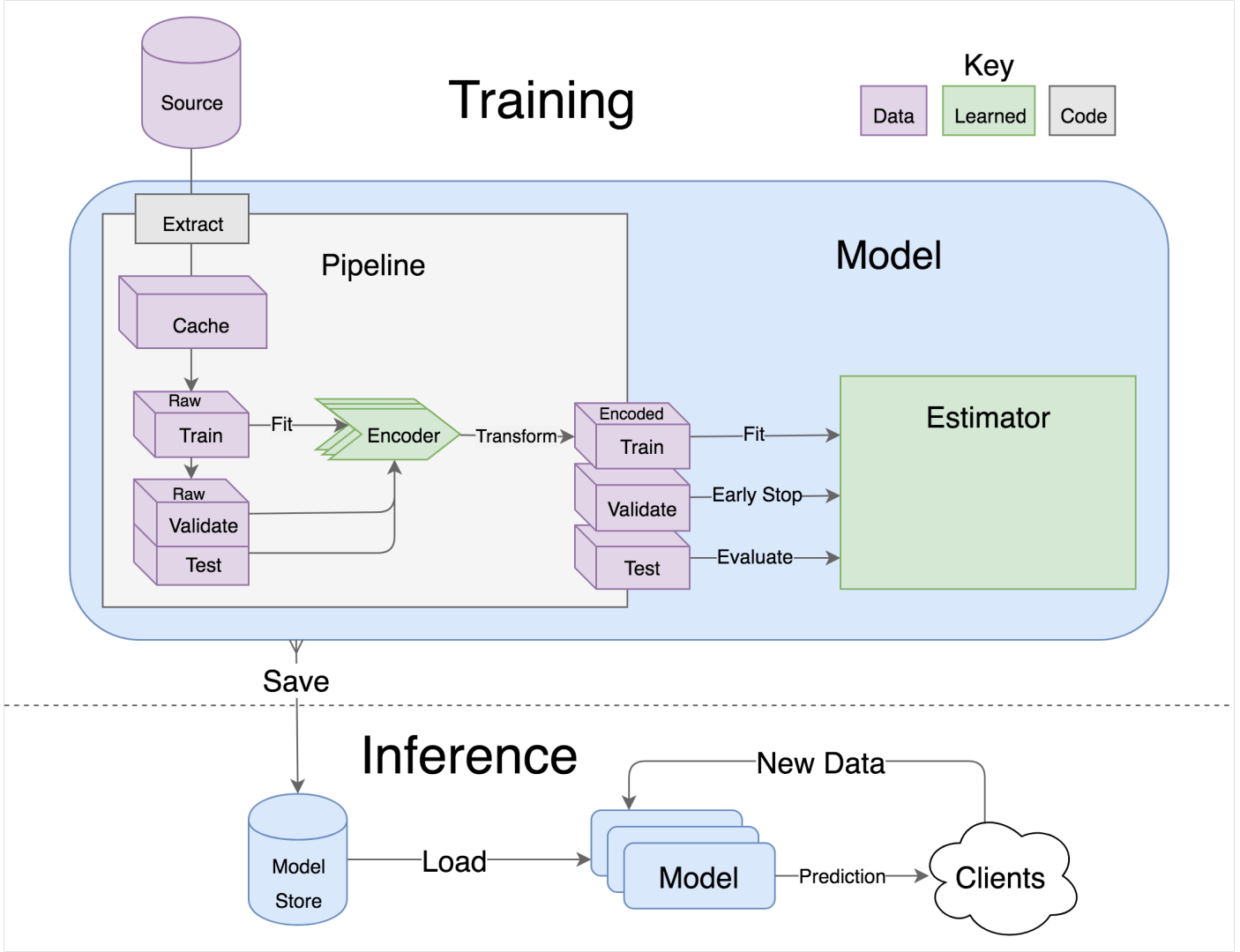
loss_prevention/models/credit_card_disputes.py

```
from loss_prevention.pipelines.credit_card_diputes import Pipeline

class DeepLearning(lore.models.keras.Base):
  def __init__(self):
    super(DeepLearning, self).__init__(
      pipeline=Pipeline(),
      estimator=lore.estimators.keras.BinaryClassifier()
    )

  @timed(logging.INFO)
  def predict(self, dataframe):
    data = self.pipeline.get_data(delivery_id=dataframe.delivery_id)
    return self.estimator.predict(data)
```

Lore Server

```
$ lore server &

loss_prevention in development on montanalow@localhost
Using TensorFlow backend.
* Serving Flask app "lore.www"



$ curl http://localhost:5000/delivery_disputes.DeepLearning/predict.json -d "delivery_id=123"

True
```

instacart


Training
Source
Key
Data
Learned
Code
Extract
Pipeline
Model
Cache
Raw
Train
Fit
Encoder
Transform
Encoded
Train
Fit
Estimator
Raw
Validate
Early Stop
Validate
Test
Evaluate
Test
Save
Inference
New Data
Model
Store
Load
Model
Prediction
Clients

instacart

Transformers

- GeoIP
- Distance
- DateTime
- String
- EmailDomain
- AreaCode
- Log/PlusOne
- ...
- NameAge
- NameSex
- NameFamilial
- NamePopulation

Encoders

- Norm
- Quantile
- Discrete
- Boolean
- Enum
- Unique
- Token
- Glove
- MiddleOut
- Equals

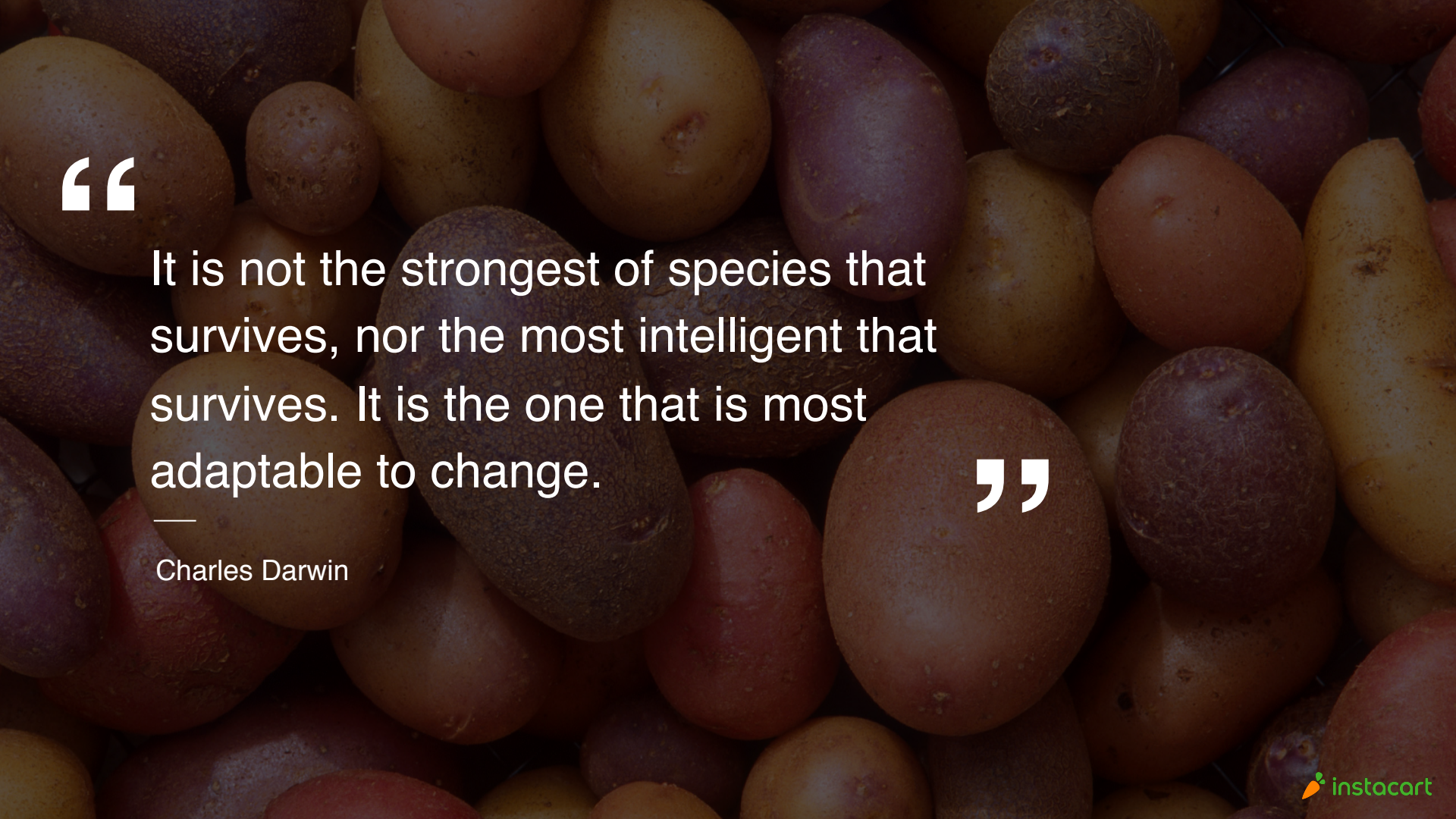
Algorithms

- Keras/Tensorflow
- XGBoost
- SciKit Learn

WE’RE HIRING!

montana@instacart.com

instacart


“
It is not the strongest of species that survives, nor the most intelligent that survives. It is the one that is most adaptable to change.
”
—
Charles Darwin
instacart