AI Do's and Don'ts

PROLEGO

Prolego delivers practical AI engineering services to Fortune 1000 companies

Getting Real

A firehose of practical advice for your AI journey

Today’s Agenda

- Strategy
- Prototype
- Deployment

Here it comes!

Decide what is relevant for you.

Strategy

What can AI do?

How do we start?

BECOME AN AI COMPANY IN 90 DAYS
The complete guide for understanding AI, identifying opportunities, and launching your first product
KEVIN DEWALT
Produced by: Russ Rands

Download an ecopy at book at www.prolego.com

AI Product Patterns

Pattern 1
Computer Vision

- *What is in this image?*
- *Tell me when something changes*

Pattern 2
Natural Language Processing (NLP)

- *Transcribe, translate his speech*
- *Notify analyst if we should review*

Pattern 3
Next-in-Sequence Predictions

- *Predicting future sales of an item based on past sales*
- *Predicting which users will buy*
- *Detecting fraudulent credit card transactions*

Presenting NLP in a New Way: Adventures in AI

Inspired by GE's use of comic books when introducing new technologies in the 1940s and 1950s, we created our own comic book, Adventures in AI.

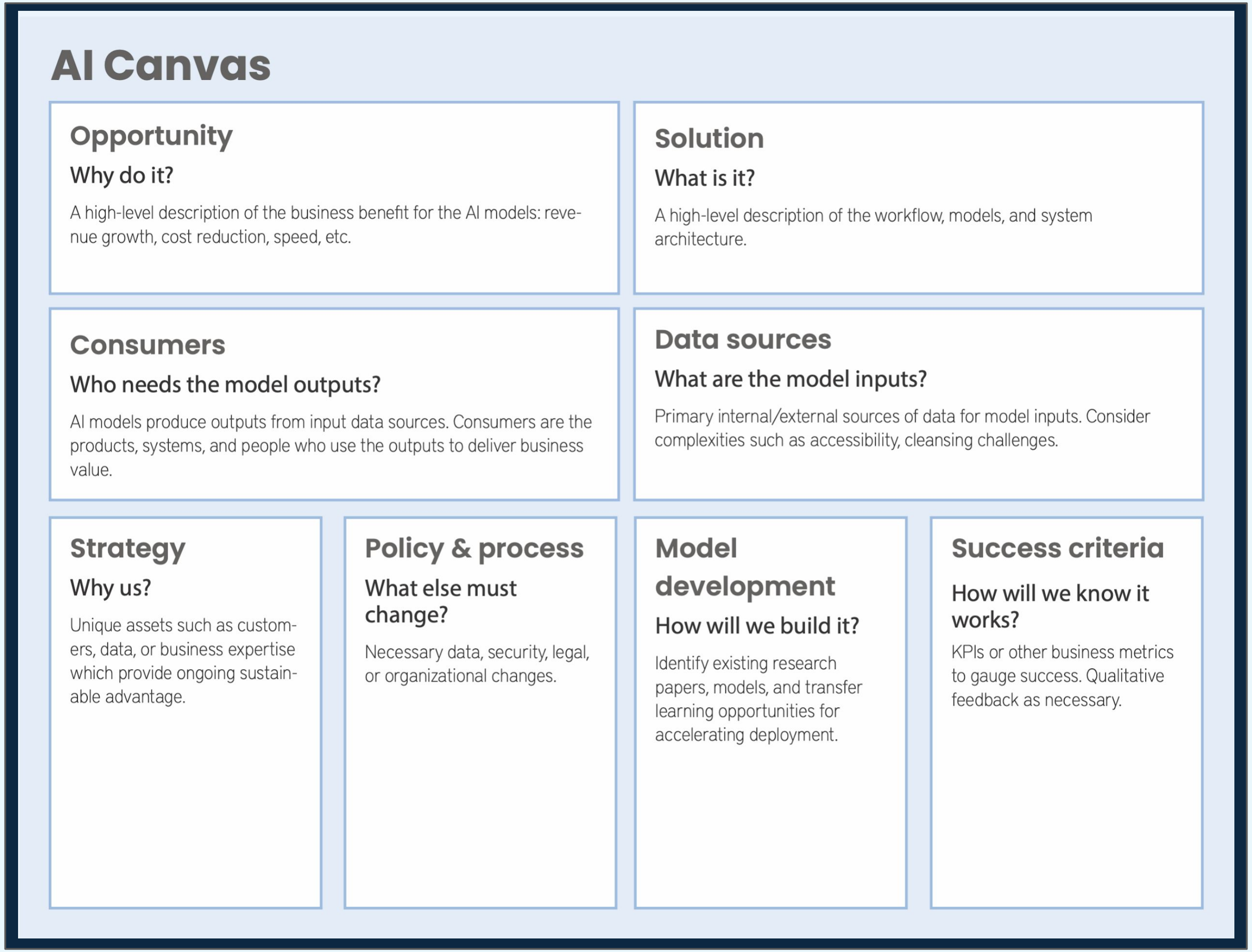
1950

ADVENTURES in Electricity
NUMBER ONE
YOU'D BETTER PEDAL HARDER, JOHNNY!
SCIENTISTS HAVE LEARNED TO TAP THE STORED ENERGY OF THE SUN...TO FLASH IT THOUSANDS OF MILES IN A FRACTION OF A SECOND...AND TO PERFORM MODERN MIRACLES THAT PUT TO SHAME THE FABLED MAGIC OF THE ARABIAN NIGHTS. WHAT IS THIS STRANGE AND MARVELOUS FORCE? HOW DO WE CONTROL THE MONSTROUS POWER WE UNLOOSE? THE ANSWER'S IN THIS EXCITING TALE OF
"HOW MAGIC IS BORN ...AND HOW IT TRAVELS!"
ADVENTURE SERIES
Prepared for
GENERAL ELECTRIC COMPANY
PICTORIAL MEDIA, INC
THE GENERATION OF ELECTRICITY

2020

Download a free copy at
www.prolego.com/adventures-in-ai

Project Selection



AI Canvas

Opportunity

Why do it?

A high-level description of the business benefit for the AI models: revenue growth, cost reduction, speed, etc.

Solution

What is it?

A high-level description of the workflow, models, and system architecture.

Consumers

Who needs the model outputs?

AI models produce outputs from input data sources. Consumers are the products, systems, and people who use the outputs to deliver business value.

Data sources

What are the model inputs?

Primary internal/external sources of data for model inputs. Consider complexities such as accessibility, cleansing challenges.

Strategy

Why us?

Unique assets such as customers, data, or business expertise which provide ongoing sustainable advantage.

Policy & process

What else must change?

Necessary data, security, legal, or organizational changes.

Model development

How will we build it?

Identify existing research papers, models, and transfer learning opportunities for accelerating deployment.

Success criteria

How will we know it works?

KPIs or other business metrics to gauge success. Qualitative feedback as necessary.

Prototype

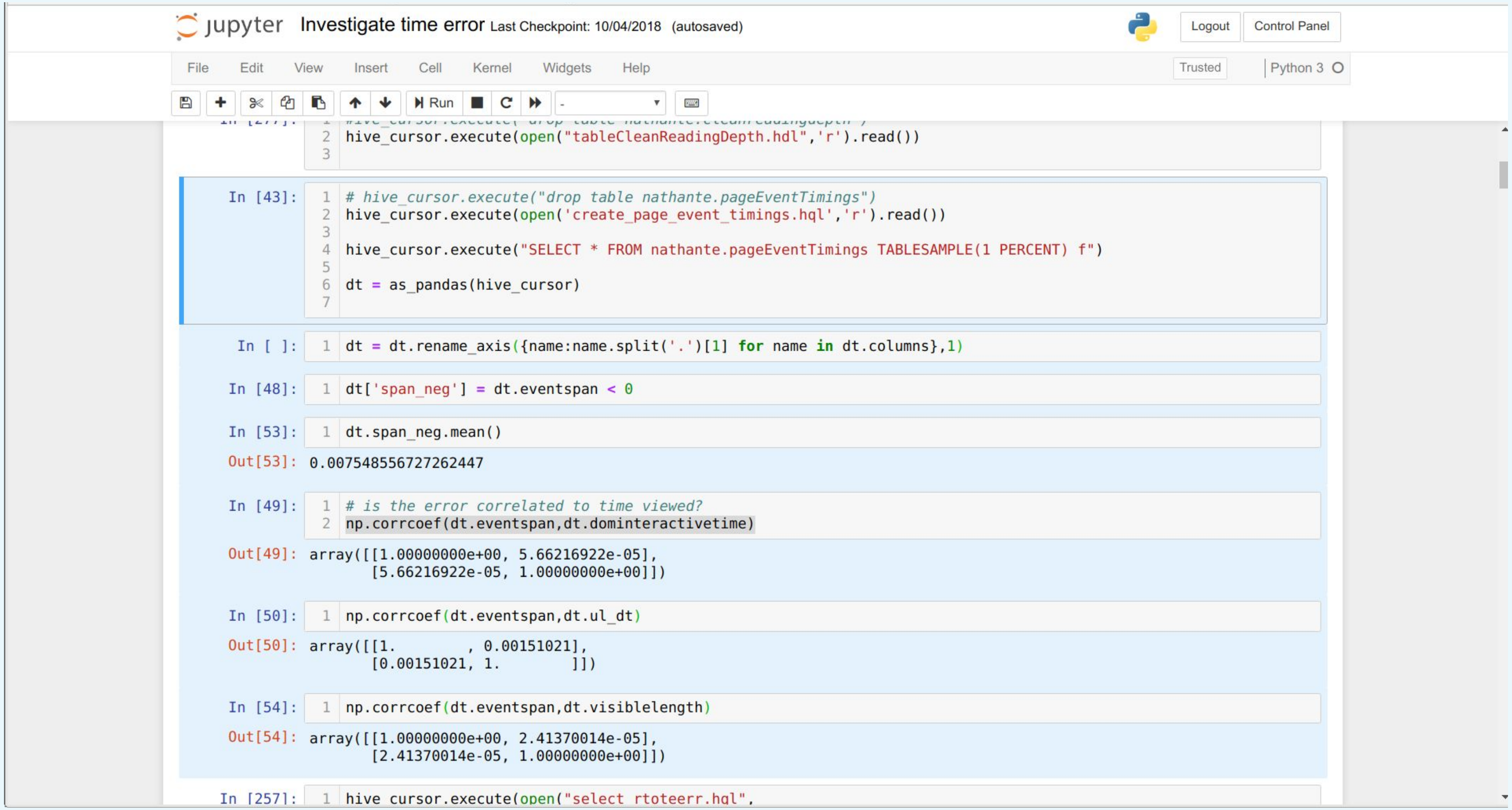
Get Real!

Budget: *creating capability, not immediate ROI*

Team: *current talent before recruiting*

Data: *don’t delay due to “data issues”*

Evaluate impact on your data



jupyter Investigate time error Last Checkpoint: 10/04/2018 (autosaved)
Logout
Control Panel
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run

```
    hive_cursor.execute(open("tableCleanReadingDepth.hdl",'r').read())

In [43]:
    # hive_cursor.execute("drop table nathante.pageEventTimings")
    hive_cursor.execute(open('create_page_event_timings.hql','r').read())

    hive_cursor.execute("SELECT * FROM nathante.pageEventTimings TABLESAMPLE(1 PERCENT) f")

    dt = as_pandas(hive_cursor)

In [ ]:
    dt = dt.rename_axis({name:name.split('.')[1] for name in dt.columns},1)

In [48]:
    dt['span_neg'] = dt.eventspan < 0

In [53]:
    dt.span_neg.mean()

Out[53]: 0.007548556727262447

In [49]:
    # is the error correlated to time viewed?
    np.corrcoef(dt.eventspan,dt.dominteractivetime)

Out[49]: array([[1.00000000e+00, 5.66216922e-05],
                [5.66216922e-05, 1.00000000e+00]])

In [50]:
    np.corrcoef(dt.eventspan,dt.ul_dt)

Out[50]: array([[1.        , 0.00151021],
                [0.00151021, 1.        ]])

In [54]:
    np.corrcoef(dt.eventspan,dt.visiblelength)

Out[54]: array([[1.00000000e+00, 2.41370014e-05],
                [2.41370014e-05, 1.00000000e+00]])

In [257]:
    hive_cursor.execute(open("select_rtoteerr.hql",
```

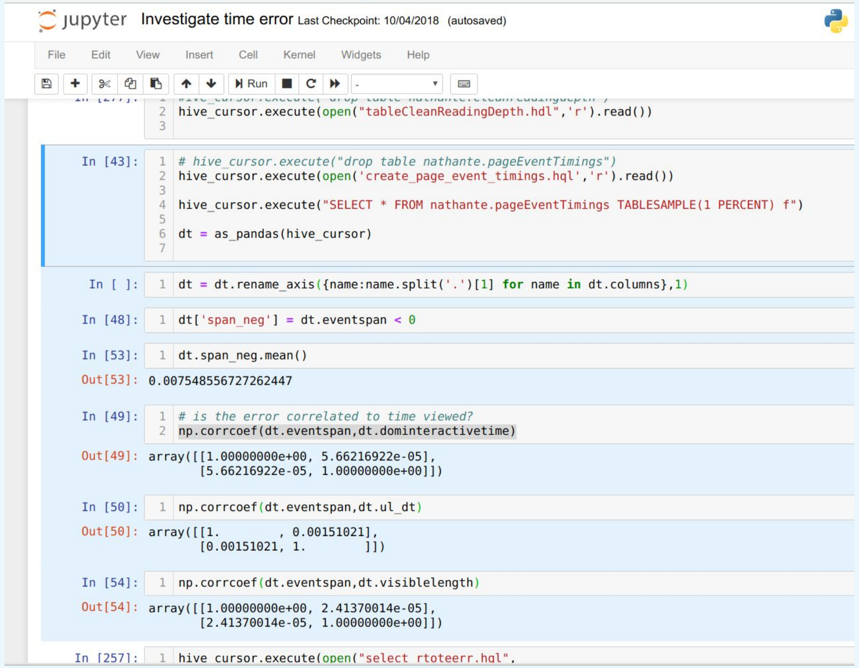
Tips

- Time-box to 4 months max!
- Work with engaged business partners
- Plan for deployment on day 0

Deployment

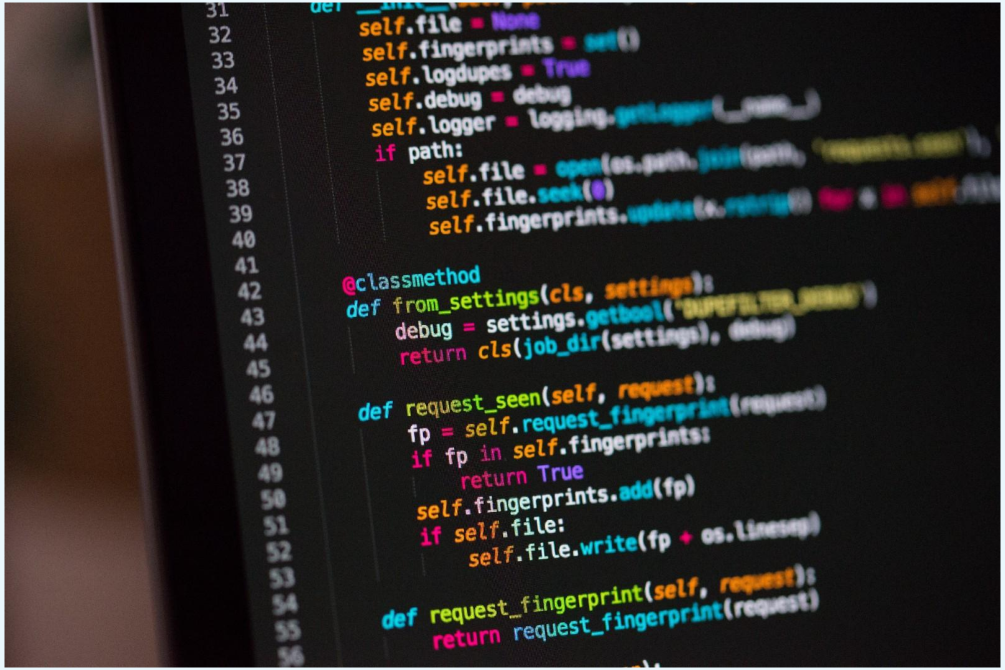
That light … is indeed a train

From Notebooks to Production Code



jupyter
Investigate time error Last Checkpoint: 10/04/2018 (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run

```
In [277]: #hive_cursor.execute("drop table nathante.cleanReadingDepth")
          hive_cursor.execute(open("tableCleanReadingDepth.hdl",'r').read())

In [43]:  # hive_cursor.execute("drop table nathante.pageEventTimings")
          hive_cursor.execute(open('create_page_event_timings.hql','r').read())

          hive_cursor.execute("SELECT * FROM nathante.pageEventTimings TABLESAMPLE(1 PERCENT) f")

          dt = as_pandas(hive_cursor)

In [ ]:   dt = dt.rename_axis({name:name.split('.')[1] for name in dt.columns},1)

In [48]:  dt['span_neg'] = dt.eventspan < 0

In [53]:  dt.span_neg.mean()

Out[53]:  0.007548556727262447

In [49]:  # is the error correlated to time viewed?
          np.corrcoef(dt.eventspan,dt.dominteractivetime)

Out[49]:  array([[1.00000000e+00, 5.66216922e-05],
                 [5.66216922e-05, 1.00000000e+00]])

In [50]:  np.corrcoef(dt.eventspan,dt.ul_dt)

Out[50]:  array([[1.        , 0.00151021],
                 [0.00151021, 1.        ]])

In [54]:  np.corrcoef(dt.eventspan,dt.visiblelength)

Out[54]:  array([[1.00000000e+00, 2.41370014e-05],
                 [2.41370014e-05, 1.00000000e+00]])

In [257]: hive_cursor.execute(open("select_rtoteerr.hql",
```



```
    def __init__(self, [illegible]
        self.file = None
        self.fingerprints = set()
        self.logdupes = True
        self.debug = debug
        self.logger = logging.getLogger(__name__)
        if path:
            self.file = open(os.path.join(path, [illegible]
            self.file.seek(0)
            self.fingerprints.update(x.rstrip() [illegible]

    @classmethod
    def from_settings(cls, settings):
        debug = settings.getbool([illegible])
        return cls(job_dir(settings), debug)

    def request_seen(self, request):
        fp = self.request_fingerprint(request)
        if fp in self.fingerprints:
            return True
        self.fingerprints.add(fp)
        if self.file:
            self.file.write(fp + os.linesep)

    def request_fingerprint(self, request):
        return request_fingerprint(request)
```

Closing Thoughts

- Think defense.
- Think long term.
- Think capabilities - not solutions.