Machine Learning Projects

The Same and The Different

Bijay Gurung / @bglearning
Python User Group Nepal Meetup #14
Outline

1. `machine_learning()` # What and Why
2. `compare(normal_prog, ml_prog)`
3. for perspective in perspectives:
   for topic in perspective:
     discuss(topic.same)
     discuss(topic.different)
What and Why Machine Learning?
Programming
What and Why Machine Learning?

How can we detect cats in an image?
Cats!
“So many variations!” :|
ML:
“So much data!” :D
ARTIFICIAL INTELLIGENCE

MACHINE LEARNING

Building **intelligent** machines.

Building **intelligent** machines that **learn** (from data)
Why Machine Learning?

Because for a lot of problems we can’t explicitly define the solution.
ML Programming
ML Programming
ML Programming

Program space

Software 1.0

Software 2.0

(optimization)

Program complexity

Source: Software 2.0
Approaches
Approaches | Extremes

Input → Computation → Output

Program

Input → Computation → Output

Program

Input → Computation → Output

Input → Output
Approaches | Extremes

“They are doing the same thing: computation”
Approaches | Extremes

“ML is different; it’s experiments, it’s science!”
ML Project | Approaches

- The Software Engineering approach: Treat ML projects like any other software project.

- The Academic approach: Not think of ML projects as software projects.
ML Project | Approaches

- The Software Engineering approach: Treat ML projects like any other software project.
  > Needs to pay heed to what’s different.

- The Academic approach: Not think of ML projects as software projects.
ML Project | Approaches

● The Software Engineering approach: Treat ML projects like any other software project.
  > *Needs to pay heed to what’s different.*

● The Academic approach: Not think of ML projects as software projects.
  > *Needs to pay heed to what’s the same.*
Normal vs ML Projects

Perspectives:

<table>
<thead>
<tr>
<th>Project/Product Owner</th>
<th>Project Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer</td>
<td>DevOps</td>
</tr>
<tr>
<td>Project/Product Owner</td>
<td>Project Manager</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Developer</td>
<td>DevOps</td>
</tr>
</tbody>
</table>
Requirements | The Same

Both have the same goals:

Solve Problems, Add Value
Requirements
| The Same

Why are we doing this?

We're bored is all!
Requirements

| The Different

More questions to ask:

● Is ML necessary?

● What’s the expectation for the ML component?
Requirements
| The Different

When a user takes a photo, the app should check whether they're in a national park...

Sure, easy GIS lookup... gimme a few hours.

... and check whether the photo is of a bird.

I'll need a research team and five years.

In CS, it can be hard to explain the difference between the easy and the virtually impossible.
Feasibility
| The Same

Needs to be feasible from various perspectives:

- Technical
- Economic
- Legal
- Operational
- Schedule
Feasibility

| The Different

For ML, need to look at technical feasibility for the ML component:

- Do we have the necessary data? If not, can we acquire it?
- What’s the state-of-the-art in this area?
<table>
<thead>
<tr>
<th>Project/ Product Owner</th>
<th>Project Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer</td>
<td>DevOps</td>
</tr>
</tbody>
</table>
Iterative Approach

| The Same

- Iterate! Iterate! Iterate!

---

**ANALYZE**
Find performance bottleneck

**SELECT APPROACH**
Craft list of experiments

**MEASURE**
Build opinionated dashboard

**IMPLEMENT**
Code models and pipeline

Source:
ML Engineering Loop
Iterative Approach

| The Same

- Have intermediate outputs
Iterative Approach

| The Different

- What are “intermediate outputs” in ML projects?
Iterative Approach
| The Different

The Data Value Pyramid.
Intermediate outputs as we climb it.

Source: Agile Data Science
Task formulation & Planning | The Same

- Need to have some level of planning: a systematic approach
- Formulate tasks with clear what, why, and how
Task formulation & Planning

| The Different |

- Put in more areas for flexibility in the process.

*Because: ML projects have a big experimental component.*
Task formulation & Planning
| The Different

- Put in more areas for flexibility in the process.
  A lot of ML projects are experimental.

Eg: Kanban vs Sprints
Task formulation & Planning | The Different

- Put in more areas for flexibility in the process.
  A lot of ML projects are experimental.

Do away with User Stories?

As a ML Developer, I want to increase the accuracy of the model by 5% so that it does what it is supposed to
Task formulation & Planning

| The Different

- Try to be more precise about what “Done” means
Try to be more precise about what “Done” means.

**PM:** “Is the task Done?”

**MLE:** “Yes... umm... no. I’ve a feeling it needs more tuning... “
<table>
<thead>
<tr>
<th>Project/Product Owner</th>
<th>Project Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Developer</strong></td>
<td><strong>DevOps</strong></td>
</tr>
</tbody>
</table>
Version Control
| The Same

- Version Control your code!
Version Control
| The Different

- Version Control your code!
  Also: data, model, configs.
Version Control
| The Different

- Version Control your code!
  Also: data, model, configs.

My Drive  ➔  Data for Project "World Domination"

Folders

- Version1
- Version2
- Version42
- Version51 😞
Version Control
| The Different

- Version Control your code!
  Also: data, model, configs.
Version Control | The Different

- Version Control your code!
  Also: data, model, configs.
Version Control | The Different
Version Control

| The Different

- Think about reproducibility
Version Control

| The Different

- Multi-experimental-branch Tendency

Formulate a policy
Flexibility
| The Same

- Balance between future-proofing and getting things done
Flexibility
| The Same

- Balance between future-proofing and getting things done
Flexibility
| The Different

- A lot of the flexibility is for experimentation rather than for changes in product features and implementation.
Flexibility | The Different

- Which one should I do?

```python
class MyModel(Model):
    def __init__(self):
        self.input_embedding = Embedding(100)
        self.encoder = LSTM(100, 200)
        self.my_novel_bits = ...
```

```python
class MyModel(Model):
    def __init__(self,
        input_embedding: TextFieldEmbedder,
        encoder: Seq2SeqEncoder):
        self.input_embedding = input_embedding
        self.encoder = encoder
        self.my_novel_bits = ...
```

On the parts that aren’t what you’re focusing on, you start simple. Later add ELMo, etc., without rewriting your code.

Source: Writing Code for NLP Research, AllenNLP
Components & Abstractions

| The Same

- Break into cohesive, de-coupled components
Components & Abstractions
| The Same

- Break into cohesive, de-coupled components; ML is small part of a bigger whole

Source: Hidden Technical Debt in Machine Learning
Components & Abstractions
| The Different

- Components can “entangle” in weird ways.
Components & Abstractions | The Different

- Components can “entangle” in weird ways.

**The CACE principle:**
*Changing Anything Changes Everything*
Components & Abstractions

| The Different |

- Components can “entangle” in weird ways.

**The CACE principle:**
*Changing Anything Changes Everything*

*Why?*

Dependence on both code and data
Testing | The Same

- Standard tests (unit tests, integration tests, etc) important
Testing | The Same

- Standard tests (unit tests, integration tests, etc) important
- Focus on critical and/or riskier parts
Testing
| The Different

More tests necessary!

Source: A rubric for ML production readiness
Good Practices
| The Same

- Use of linters, style-guides, etc
- Readable, well-commented code
- Documentation
Good Practices

| The Different |

- Data Documentation

<table>
<thead>
<tr>
<th>Table</th>
<th>Table Notes</th>
<th>Field</th>
<th>Definition</th>
<th>Example Value</th>
<th>Field Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>website_actions</td>
<td>Tracks user actions on the website; updated nightly</td>
<td>user_id</td>
<td>User ID</td>
<td>55555</td>
<td>Tie to user table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>action</td>
<td>User action on website</td>
<td>watched_video</td>
<td>Values: clicked_button, watched_video, signed_up_for_thing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>object_id</td>
<td>Object a user interacted with</td>
<td>222</td>
<td>Tie to object table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>timestamp</td>
<td>Date and time of user interaction</td>
<td>2018-01-02 1:01:31</td>
<td>Data appears from 1/1/18 to present</td>
</tr>
</tbody>
</table>

Source: Field Notes: Building Data Dictionaries
Good Practices
| The Different

- Experimental logs

Source: Writing Code for NLP Research, AllenNLP
<table>
<thead>
<tr>
<th>Project/Product Owner</th>
<th>Project Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer</td>
<td>DevOps</td>
</tr>
</tbody>
</table>
Deployment & Maintenance
| The Same

- Need Continuous Integration, Build Automation, Monitoring, etc
Deployment & Maintenance | The Different

- Also need to monitor data, model performance, etc.
How is MLOps different from DevOps?

- Data/model versioning != code versioning - how to version data sets as the schema and origin data change
- Digital audit trail requirements change when dealing with code + (potentially customer) data
- Model reuse is different than software reuse, as models must be tuned based on input data / scenario.
- To reuse a model you may need to fine-tune / transfer learn on it (meaning you need the training pipeline)
- Models tend to decay over time & you need the ability to retrain them on demand to ensure they remain useful in a production context.

Source: MLOps, Microsoft
Deployment & Maintenance

| The Different

Essence:
Not just code artefacts, overall application performance
Also, data, model artefacts, and their operational behavior
Thinking Meta
Where is the difference coming from?

ML projects:

- Depend on/tightly coupled with data (in addition to code & infrastructure)
- Can be thought of as being “Two phase”
- Is usually highly experimental
Where is the similarity coming from?

ML Projects:

- Have ML component as just one part of a bigger system that’s meant to solve a problem.
Wrapping Up
What?

Machine Learning Projects are both same as and different than “normal” software projects.
So What?

When doing ML projects, it’s best to:

- Use Software Engineering principles
- And also think about the unique challenges of ML projects

Balance is key
So What?

When doing ML projects, it’s best to:

- Use Software Engineering principles
- And also think about the unique challenges of ML projects

Balance is key

Now What? Up to you :)