AI/ML session.

* What is a Machine Learning Expert?
	+ Can be someone training in ML theory.
		- Theory knowledge can help determine algorithm to use.
	+ Someone to develop and monitor deployed system, and design it in a way that it can be updated by users.
	+ ML Grad students or postdocs can be your ML expert.
	+ It’s possible to use tools without formal training. Don’t be scared to try. 90% of the time there are less complex solutions available instead of ML.
	+ if non-ML expert is involved (physicist) watch for only using favorite or minimal tool set. One hammer used for everything
* How do you get the right data format?
	+ ML trained should have knowledge to transform from whatever you give them
	+ ML expert will need to gain knowledge and work closely with experts of the accelerator systems, data collection systems, and domains to understand the data and determine what is “normal” and what is “weird”.
	+ Data dictionary can help.
	+ Data format can vary but best collection format may be serial.
	+ Students without industry experience often only trained on sanitized data sets. They may need to learn how to clean and collect data @ accelerators.
	+ Unfamiliar experts not understanding operations of system. Slow drift happens in the data (data drift). Model adjustments will be needed in operations.
* Deployed systems
	+ Hardware point – do you need IT for the system? Not always. Where does it go and who owns it when it’s deployed? Many of the current examples are not yet in a final state for operations (they still require babysitting by ML experts)
	+ How much support from IT? Various depending on the scope of the problem being solved and the specifications of the accelerator’s I/O.
	+ MaxIV – using virtual tango on device server to eliminate hardware. With response times at about 1 second.
	+ Alex Shenka has work on optimization algorithm with proof of concept at different accelerator facilities.
	+ RF cavity breakdown prediction to power down before a quench happens.
	+ SLAC XOpt for Bayesian optimization of linac.
* How do you begin to trust an ML solution?
	+ Before deploying a system, you need to gain confidence in its predictions. Plan a phased approach to deployment to evaluate before you decide to deploy automated actions.
	+ You can also have a dual process of side-by-side models. One is live and its performance is evaluated on current live data, the second model is running either on a simulation, or just concurrently without taking action on the system. This secondary model can be trained on recent data and it’s predictions monitored, when it’s predictions do a better job than the live model, switch them out. This can have various levels of human oversight as confidence and tests are built into the system.
	+ First develop an auxiliary ML solution for a “problem” that is **already solved** to get feet wet.
		- FNAL & ORNL: Booster PID loop, muon slow spill extraction -> neural network or boosted decision tree on fpga
		- Linac auto-tune scan applications -> XOpt baysiean optimization program
	+ “Explainable AI” is a field of study developing tools to understand why deep learning neural networks make the decisions they do. This is important so you can evaluate if your model is emphasizing the right data, it will inform hoy you might revise a model to give better predictions.
	+ Keep “traditional system” in place, and have an optional “off” button for the ML model for operators.
* How do I get started?
	+ Get data out of instrumentation devices/data logger.
		- There are strategies to try to line up different device data in time / by pulse.
	+ First start with exploratory data analysis (EDA). Make plots. Look for trends and correlations.
		- Tools:
			* Python 3: Numpy, Pandas, MatPlotLib
			* Grafana GUI for exploration
			* Others from business industry: Tableau $$$
	+ Label good or bad (for example for supervised learning techniques)
	+ Decide if it’s a classifying problem, or regression problem (predict a classification, or predict a float/continuous variable. And what metric you want to optimize? How much is “good enough?”
	+ Try out an ML method that matches this type of problem (searchable on google, or ask advice from a local expert), maybe fancy cutting edge methodologies are available on GitHub.
		- Tools:
			* Python3: SciKitLearn’s boosted decision trees, deep learning: Keras/Tensorflow, Pytorch