SOCIETY OF ACTUARIES

2019 ANNUAL MEETING & EXHIBIT

October 27-30
Toronto, Canada

Session 190: Getting Started with Predictive Analytics: Kaggle Competitions

2019 ANNUAL MEETING & EXHIBIT

Getting Started with Predictive Analytics: Kaggle Competitions

Presenter: MARIA WELLEN, ASA

Moderator: NATE POHLE, FSA, MAAA, CERA

10/30/2019

SOCIETY OF ACTUARIES®

SOCIETY OF ACTUARIES
Antitrust Compliance Guidelines

Active participation in the Society of Actuaries is an important aspect of membership. While the positive contributions of professional societies and associations are well-recognized and encouraged, association activities are vulnerable to close antitrust scrutiny. By their very nature, associations bring together industry competitors and other market participants.

The United States antitrust laws aim to protect consumers by preserving the free economy and prohibiting anti-competitive business practices; they promote competition. There are both state and federal antitrust laws, although state antitrust laws closely follow federal law. The Sherman Act, is the primary U.S. antitrust law pertaining to association activities. The Sherman Act prohibits every contract, combination or conspiracy that places an unreasonable restraint on trade. There are, however, some activities that are illegal under all circumstances, such as price fixing, market allocation and collusive bidding.

There is no safe harbor under the antitrust law for professional association activities. Therefore, association meeting participants should refrain from discussing any activity that could potentially be construed as having an anti-competitive effect. Discussions relating to product or service pricing, market allocations, membership restrictions, product standardization or other conditions on trade could arguably be perceived as a restraint on trade and may expose the SOA and its members to antitrust enforcement procedures.

While participating in all SOA in person meetings, webinars, teleconferences or side discussions, you should avoid discussing competitively sensitive information with competitors and follow these guidelines:

- Do not discuss prices for services or products or anything else that might affect prices
- Do not discuss what you or other entities plan to do in a particular geographic or product markets or with particular customers.
- Do not speak on behalf of the SOA or any of its committees unless specifically authorized to do so.
- Do leave a meeting where any anticompetitive pricing or market allocation discussion occurs.
- Do alert SOA staff and/or legal counsel to any concerning discussions
- Do consult with legal counsel before raising any matter or making a statement that may involve competitively sensitive information.

Adherence to these guidelines involves not only avoidance of antitrust violations, but avoidance of behavior which might be so construed. These guidelines only provide an overview of prohibited activities. SOA legal counsel reviews meeting agenda and materials as deemed appropriate and any discussion that departs from the formal agenda should be scrutinized carefully. Antitrust compliance is everyone’s responsibility; however, please seek legal counsel if you have any questions or concerns.

Presentation Disclaimer

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the participants individually and, unless expressly stated to the contrary, are not the opinion or position of the Society of Actuaries, its cosponsors or its committees. The Society of Actuaries does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented. Attendees should note that the sessions are audio-recorded and may be published in various media, including print, audio and video formats without further notice.

SOCIETY OF ACTUARIES®

Session Presented By:

Predictive Analytics and Futurism Section

Provides opportunities for actuaries to deepen their understanding of predictive analytics and emerging technologies relevant to the future of the actuarial profession and insurance industry.

Section Developed Content & Benefits

Predictive Analytics and Futurism Newsletter

Discusses futurism and the latest predictive analytics trends. Published three times a year. Digital editions now available.

SOA Meetings and Seminars

Section developed content presented during meeting sessions and seminars.

Podcasts

Expert led technical podcasts exploring the latest predictive analytics concepts and techniques.

Webcasts

Discounts on section developed webcasts. Free access to section created webcasts over one-year old.

Join the PAF Section Today! *SOA.org/PAF*

Kaggle Involvement Program

- The SOA Kaggle Involvement Program is an opportunity for actuaries to showcase their predictive modeling skills through data science competitions.
- The program period begins March 27, 2019 and ends December 31, 2019.
- Individual participants who earn the rank of "Kaggle Competitions Master" during the program period, and groups and individuals that are ranked in the top 10% of an eligible competition, will be recognized on the (1) SOA website, (2) at the 2020 SOA Annual Meeting & Exhibit, (3) and in an issue of The Actuary Magazine.
- https://www.soa.org/programs/predictive-analytics/kaggle-program/

What is Kaggle

- Kaggle is a crowdsourcing website owned by Google LLC
- Kaggle has over 19,000 public datasets and 200,000 public notebooks.
- Competitions in Kaggle are of diverse nature. Including tabular data, computer vision, and Natural Language Processing (NLP).
- Why is Kaggle important as an Actuary?

  https://www.soa.org/predictive-analytics/kaggle-program/

*https://www.kaggle.com/

Kaggle features

- Cloud based Jupyter notebooks
- Find and publish datasets
- Write Kernels to build models
- Ask questions in the Forum
- Create a team by inviting collaborators
- Free data science and machine learning courses
- Enter a competition, many of which have cash prizes
- Medal scheme progression system

How to get started with Machine Learning?

- Fastai library: https://docs.fast.ai/
- Machine learning for coders: http://course18.fast.ai/ml
- Deep Learning Course: https://course.fast.ai/

All free and open source library based on Pytorch (Python library)!!!

Important Definitions

- Train Data: Has two definitions:
  - Dataset provided by Kaggle with explanatory variables and responses.
  - Data used to fit the model after train/validation split.
  - We will refer to training data using the second definition.
- Validation Data: Data to be used in the modeling process to assess quality of the predictions. Usually ~20% of the original training data.
- Test Data: Data used at the end of the fitting cycle and on which the predictions are made.
- Public Leaderboard: Provides the team rankings while the competition is active.
- Private Leaderboard: Determines the competition standings.
- Kernel: Cloud computational environment. Supports Python and R code
- Machine Learning: Machine learning is the science (and art) of programming computers so they can learn from data[1].

References: [1] Hands-On Machine Learning with Scikit-Learn & Tensorflow Aurelien Geron 2017 Page 4

Some Problems in Machine Learning
Data Cleaning

https://www.kaggle.com/c/landmark-recognition-challenge/discussion/56436

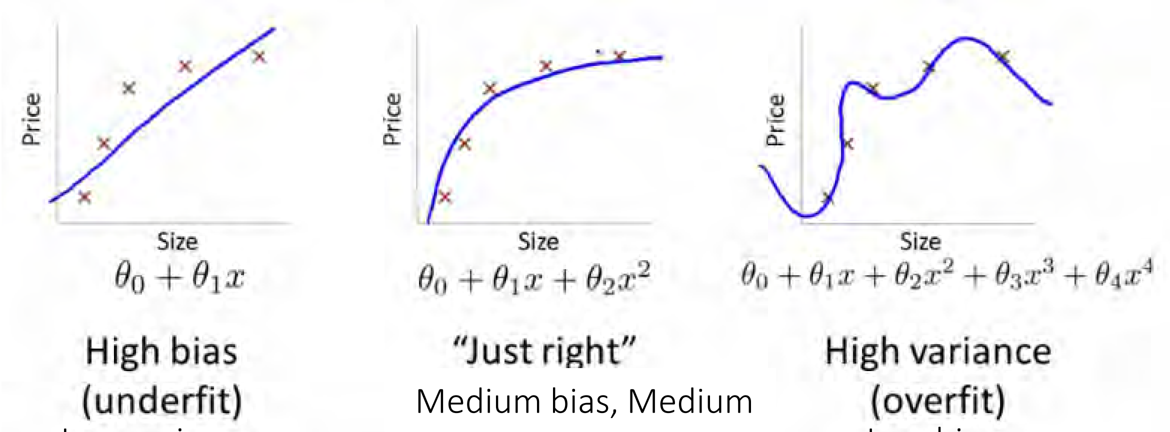
https://cloud.google.com/vision/

Refers to the treatment of inconsistent observations, missing data and outliers.

Solutions:

- Focus on data quality
- Actuarial Standard of Practice No. 23 Data Quality

Some Problems in Machine Learning
Bias variance trade off



Low variance

Medium bias, Medium variance

Low bias

Reference: https://www.kdnuggets.com/2017/11/create-good-validation-set.html

- Bias: Difference between the prediction of our model and the correct value. High bias oversimplifies the model.
- Variance: Variability of model prediction for a given data point which tells us spread of our data. High variance overfits the data.
- There is a tradeoff between a model's ability to minimize bias and variance.
- <u>Solutions:</u>
- Create an appropriate validation set by understanding the structure of the data. No overlapping data in train and validation.
- Use cross validation (to be explained).
- Create different models and average the predictions.

https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229

11

Kaggle competitions related to insurance

Competition Name	Date	Description	Link
Allstate Claim Prediction Challenge	October 2011	Predict Bodily Injury Liability Insurance claim payments based on the characteristics of the insured's vehicle.	https://www.kaggle.com/c/ClaimPredictionChallenge
Allstate Purchase Prediction Challenge	May 2014	Using a customer's shopping history, can you predict what policy they will end up choosing?	https://www.kaggle.com/c/allstate-purchase-prediction-challenge
Liberty Mutual Group - Fire Peril Loss Cost	September 2014	Predict expected fire losses for insurance policies.	https://www.kaggle.com/c/liberty-mutual-fire-peril
Liberty Mutual Group: Property Inspection Prediction	August 2015	Predict a transformed count of hazards or pre existing damages using a dataset of property information.	https://www.kaggle.com/c/liberty-mutual-group-property-inspection-prediction
Homesite Quote Conversion	February 2016	Which customers will purchase a quoted insurance plan?	https://www.kaggle.com/c/homesite-quote-conversion
State Farm Distracted Driver Detection	August 2016	Can computer vision spot distracted drivers?	https://www.kaggle.com/c/state-farm-distracted-driver-detection
Allstate Claims Severity	December 2016	How severe is an insurance claim?	https://www.kaggle.com/c/allstate-claims-severity
Porto Seguro's Safe Driver Prediction	November 2017	Predict if a driver will file an insurance claim next year	https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/

Porto Seguro's Safe Driver Prediction

Predict if a driver will file an insurance claim next year.

- Binary classification dataset (3.6% claims in the training set) - rare event
- Evaluation metric GINI = 2 * AUC - 1
- 0 random prediction, 0.5 perfect fit
- 57 variables - continuous (23) - categorical (14) - calculated (20)
- ~ 600K data points in train
- ~ 900K data points in test (to be predicted)
- Random 30/70 split in public/private leaderboard

https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/overview

Winning Approach is described in this discussion post

https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/discussion/44629#latest-632453

Overview Data Notebooks Discussion Leaderboard Rules New Topic

Michael Jahrer
1st place

1st place with representation learning

posted in Porto Seguro's Safe Driver Prediction 2 years ago

794

Thanks to Porto Seguro to provide us with such a nice, leakage-free, time-free and statistical correct dataset.
A nice playground to test the performance of everything, this competition was stat similar to otto, like larger testset than train, anonymous data, but differ in details.
I wanna dive straight into solution. Its a blend of 6 models. 1x lightgbm, 5x nn. All on same features, I just removed *calc and added 1-hot on *cat.
All neural nets are trained on denoising autoencoder hidden activations, they did a great job in learning a better representation of the numeric data. lightgbm on raw data.
Nonlinear stacking failed, simple averaging works best (all weights=1). Thats the final .2965 solution.
2 single models would have been enough to win (#1 + #2 give me 0.29502 on private).
The complete list of models in the final blend:

nr	feat	Norm-alization	unsupervised		model	5-fold CV			kaggle	
			type	time[h]	type	time[h]	gini	logloss	public	private
1	f0	-	-	0	**lightgbm** objective=binary, 1400 rounds, boosting_type=gbdt, learning_rate=0.01, max_bin=255, num_leaves=31, min_data_in_leaf=1500, feature_fraction=0.7, bagging_freq=1, bagging_fraction=0.7, lambda_l1=1, lambda_l2=1	0.13	0.28843	0.15163	0.28368	0.29097
2	f0	Rank Gauss	**denoising autoencoder, deep stack** topology=221-1500r-1500r-1500r-221l, lRate=3e-3, minibatchSize=128, backend=GPU32, lRateDecay=0.995, inputSwapNoise=0.15, nEpochs=1000	5	**neural net** topology=4500-1000r-1000r-s, lRate=1e-4, lRateDecay=0.995, regL2=0.05, dropout=0.5, dropoutInput=0.1, minibatchSize=128, backend=GPU32, loglossUpdate=1, nEpochs=150	3.45	0.29036	0.15184	0.28970	0.29298
3	f0	Rank Gauss	**denoising autoencoder, deep stack** topology=221-1500r-1500r-1500r-221l, lRate=3e-3, minibatchSize=128, backend=GPU32, lRateDecay=0.995, inputSwapNoise=0.07, nEpochs=1000	5	**neural net** topology=4500-1000r-1000r-s, lRate=1e-4, lRateDecay=0.995, regL2=0.05, dropout=0.5, dropoutInput=0.1, minibatchSize=128, backend=GPU32, loglossUpdate=1, nEpochs=200	4.6	0.28942	0.15185	0.28846	0.29377
4	f0	Rank Gauss	**denoising autoencoder, deep stack** topology=221-1500r-1500r-1500r-221l, lRate=2.9e-3, minibatchSize=128, backend=GPU32, lRateDecay=0.995, inputSwapNoise=0.15, nEpochs=1000 colGroups=1	5	**neural net** topology=4500-1000r-1000r-s, lRate=1e-4, lRateDecay=0.995, regL2=0.05, dropout=0.5, dropoutInput=0.1, minibatchSize=128, backend=GPU32, loglossUpdate=1, nEpochs=76	1.8	0.29062	0.15174	0.28778	0.29265
5	f0	Rank Gauss	**denoising autoencoder, bottleneck** topology=221-15000r-15000r-3000l-15000r-15000r-221l, lRate=1e-3, minibatchSize=128, backend=GPU32, lRateDecay=0.995, inputSwapNoise=0.1, nEpochs=300	75.2	**neural net** Topology=3000-1000r-1000r-s, lRate=1e-4, lRateDecay=0.995, regL2=0.05, dropout=0.5, dropoutInput=0.1, minibatchSize=128, backend=GPU32, loglossUpdate=1, nEpochs=200	3.2	0.28904	0.15202	0.28745	0.29373
6	f0	Rank Gauss	**denoising autoencoder, deep stack** topology=221-1500r-1500r-1500r-221l, lRate=2.8e-3, minibatchSize=128, backend=GPU32, lRateDecay=0.995, inputSwapNoise=0.2, nEpochs=1000	5	**neural net** topology=4500-1000r-1000r-s, lRate=1e-4, lRateDecay=0.995, regL2=0.05, dropout=0.5, dropoutInput=0.1, minibatchSize=128, backend=GPU32, loglossUpdate=1, nEpochs=150	3.4	0.29091	0.15180	0.28856	0.29303
					linear blend, all w=1 of 1,2,3,4,5,6	0.01	0.29442	0.15159	0.29136	0.29653

Font is a bit small, you need to increase the zoom with ctrl (+).

Main ideas

Statistical design

- One hot encoding categorical variables
- Remove calculated features
- 5 Fold cross validation
- Data augmentation

Model

- Blend of one LightGBM Model and six Neural Networks
- This works because of the Central Limit Theorem.

Hardware

- 32GB RAM machine.
- GPU: GTX 1080 Ti card for all neural networks (2 years ago)

Software

C++/CUDA - impressive! We will use Python

Statistical Design

- <u>One hot encoding</u>

ps_ind_02_cat		ps_ind_02_cat_1	ps_ind_02_cat_2	ps_ind_02_cat_3	ps_ind_02_cat_4
1		1	0	0	0
2		0	1	0	0
3		0	0	1	0
4		0	0	0	1

For tree based models and Neural networks there is no assumption of independence of covariates like in GLM models.
One hot encoding works in this case because we do not know the nature of the categorical variables.
For example assume 1: honda, 2: toyota, 3: audi, 4: lexus then it is not desirable to treat this as numeric!

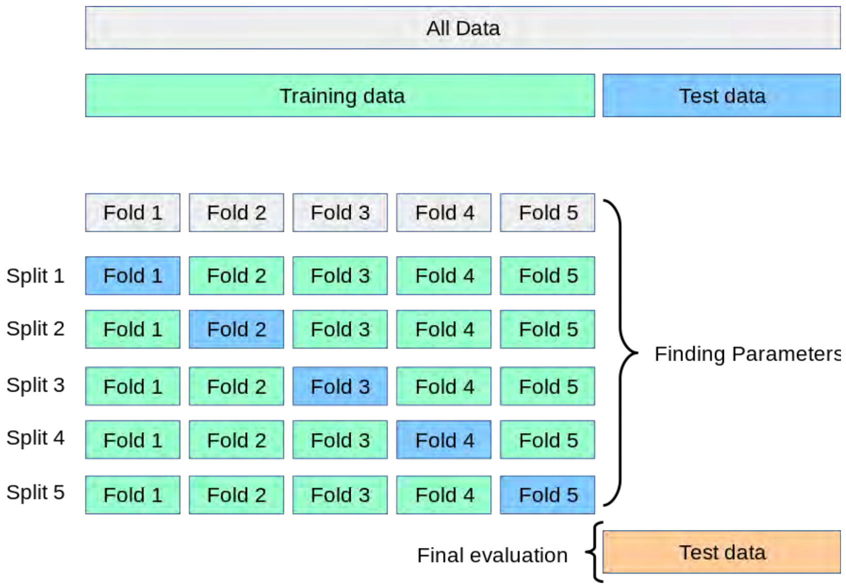
Statistical Design

- **Remove calculated features**

Reduces the search space and the complexity of the model. Only remove unimportant variables.

- **5 - fold cross validation**

Important to reduce overfitting. Standard best practice in ML/DL



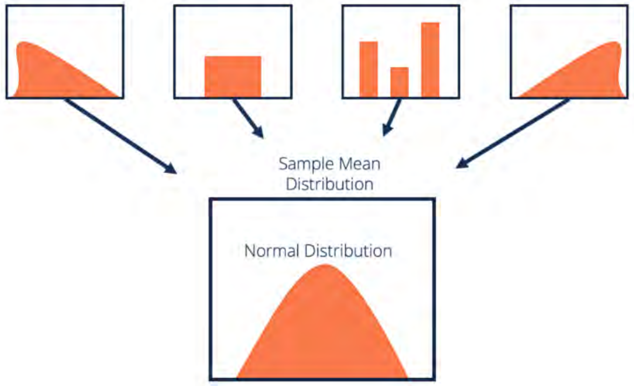
https://scikit-learn.org/stable/modules/cross_validation.html

Central Limit Theorem

- The **central limit theorem** (**CLT**) establishes that, in some situations, when *independent* random variables are added, their properly normalized sum tends toward a *normal distribution* even if the original variables themselves are not normally distributed.

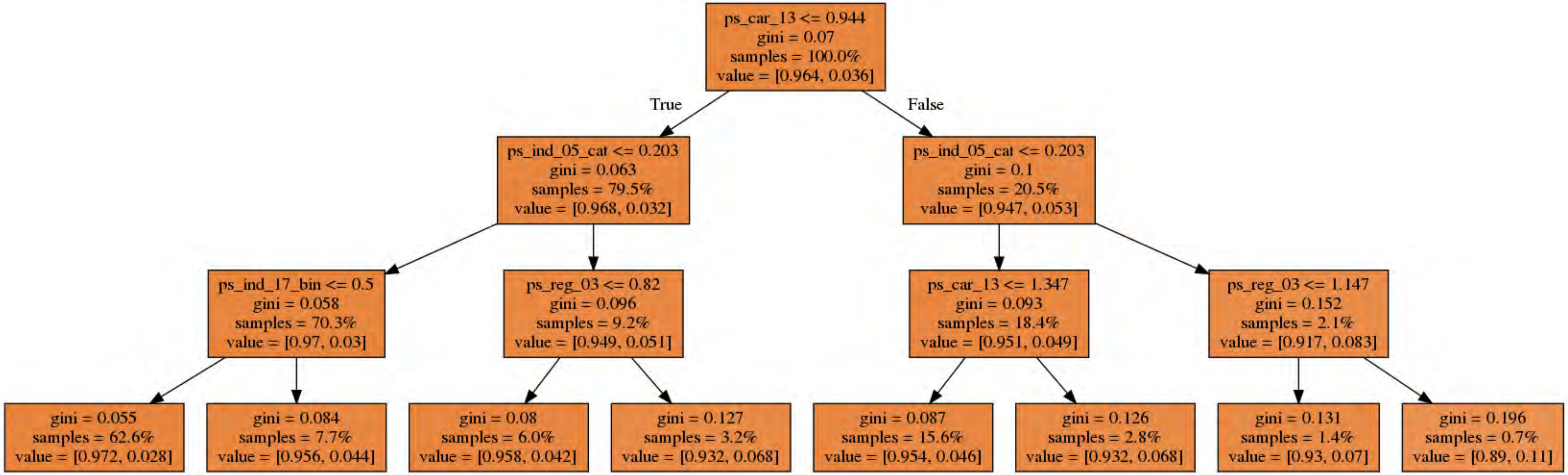
https://en.wikipedia.org/wiki/Central_limit_theorem

- This means that we can average predictions of models with *low correlation* (independent) and get a better prediction.



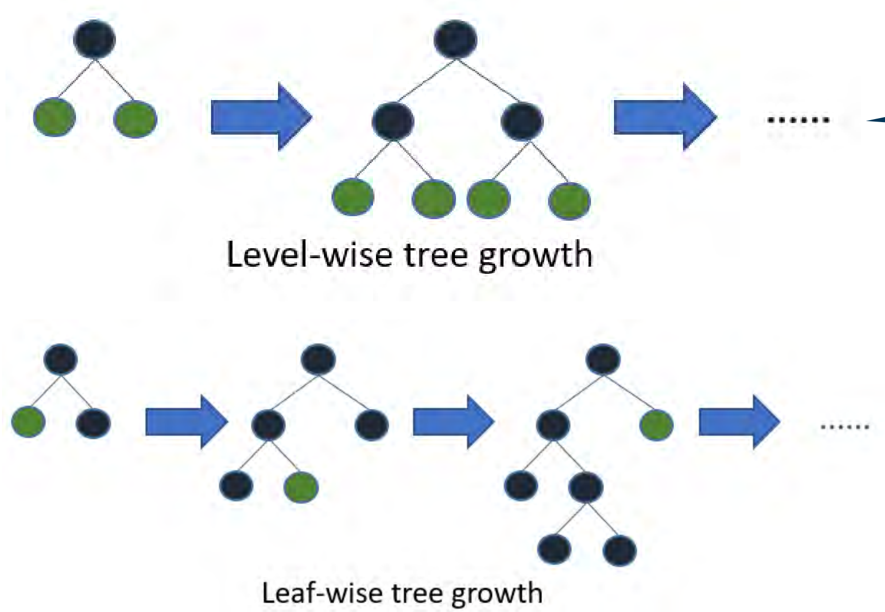
https://corporatefinanceinstitute.com/resources/knowledge/other/central-limit-theorem/

Classification Tree



Random forests, LightGBM, XGBoost, … are tree based algorithms.

LightGBM Model



Most decision tree learning algorithms grow trees by level (depth)-wise

LightGBM grows trees leaf-wise (best-first)

https://lightgbm.readthedocs.io/en/latest/Features.html

Light GBM is **sensitive to overfitting** and can easily overfit small data.
Regularization: lambda_l1 = 1. Lasso regression absolute value of the magnitude
Lambda_l2 =1. Ridge regression: square magnitude of coefficient penalty.

Porto Seguro Safe Driver Prediction https://www.kaggle.com/msmelguizo/lightgbm-demo/

1. Import Libraries for data manipulation and modeling
2. Import Train and Test Data
3. Replace -1 by missing data code
4. Set the parameters of the LigthGBM model
5. One hot encoding of the categorical variables
6. Remove the Calculated Features
7. Partition the data into 5 folds (5 fold cross-validation)
8. Fit the LightGBM model into each fold
9. Submit average of the 5 predictions on the test set

Supervised and Unsupervised learning

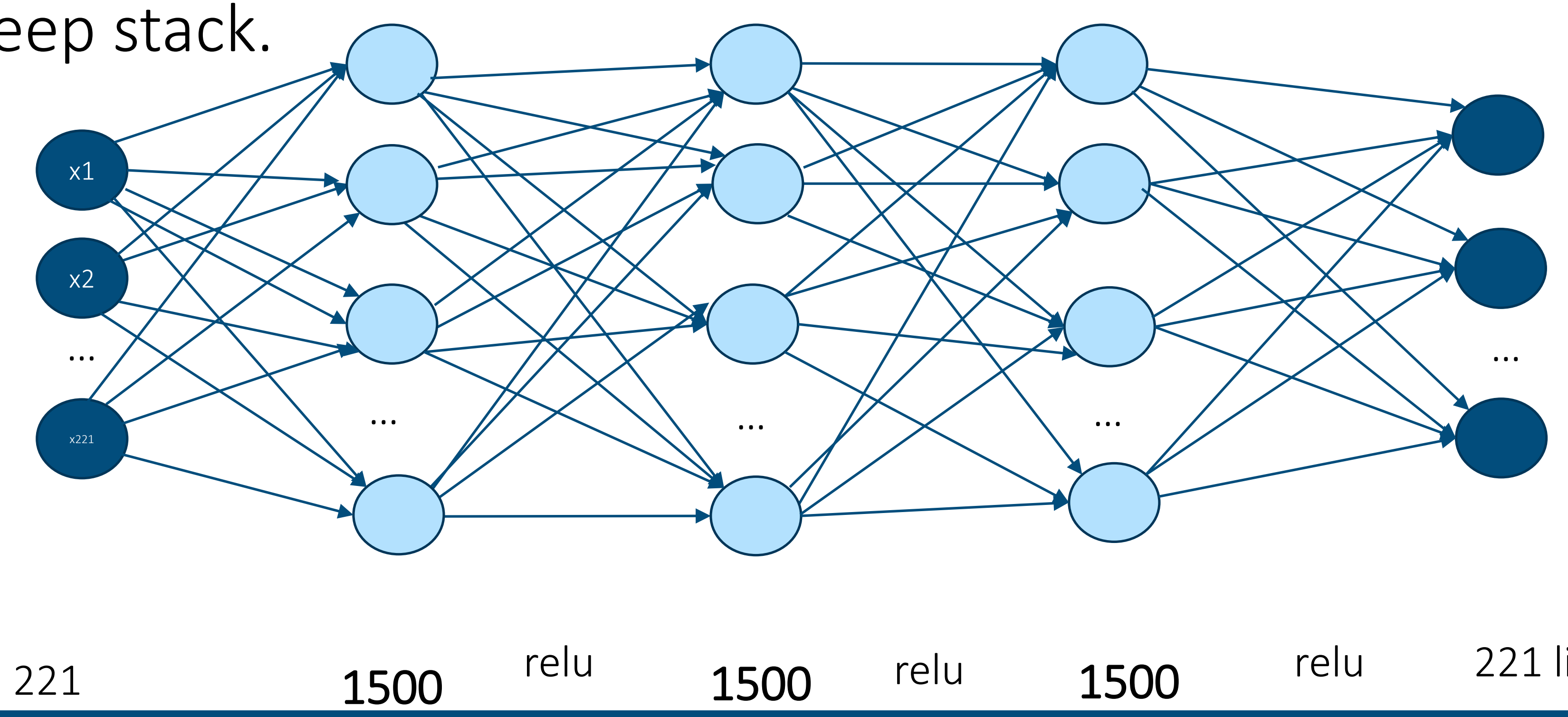
- Supervised Learning: typically done in the context of classification, when we want to map input to output labels, or regression, when we want to map input to a continuous output.
  - logistic regression
  - naive bayes
  - support vector machines
  - artificial neural networks
  - random forests, …
- Unsupervised Learning: we wish to learn the inherent structure of our data without using explicitly-provided labels.
  - k-means clustering
  - principal component analysis
  - autoencoders, …

https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d

22

Neural Networks (NN)

- Two step solution
- Step 1: Denoising Autoencoders (unsupervised): Used to generate features combining the train and test data in the input data. 221 variables (59 – 20 calculated + one hot encoded). Example here is for deep stack.

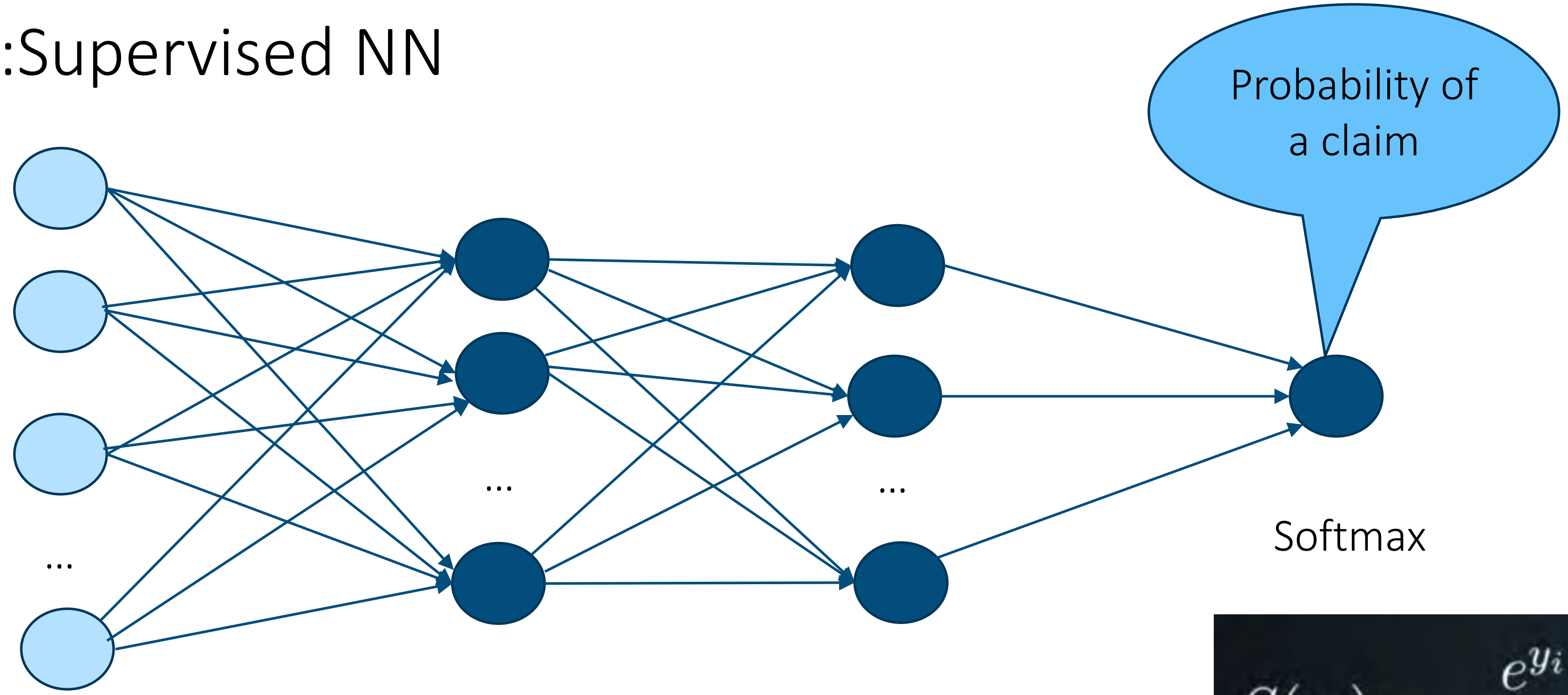


relu = max(0, x)

Objective: Minimize MSE
Lrate = 3 e-3
Minibatch size = 128
Backend=GPU32
Lrate Decay = 0.995
Input swap noise = 0.15
n epochs = 1,000

Neural Networks

- Step 2:Supervised NN



4500 = 1500 * 3 intermediate activations

1000 relu

1000 relu

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

Objective: Minimize Log-Loss
Lrate = 1e-4
Lrate Decay = 0.995
Reg L2 =0.05
Dropout = 0.5
Dropout input = 0.1
Minibatch size = 128
Backend = GPU32
N epochs = 150

4500 * 1.5 Million training data points !

Hardware

https://www.nvidia.com/en-us/geforce/20-series/

- What Is a GPU (graphics processing unit)?
- “GPU is composed of hundreds of cores that can handle thousands of threads simultaneously. The ability of a GPU with 100+ cores to process thousands of threads can accelerate some software by 100x over a CPU alone. What’s more, the GPU achieves this acceleration while being more power- and cost-efficient than a CPU.”

- NVIDIA 2080Ti (11GB) ~ $1,200
- NVIDIA 2080 (8GB) ~ $800

Both cards support half precision training (floating point precision 16 instead of 32)

https://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/

Web Based Computing Environments

Service Name	Website
Salamander	https://salamander.ai
Gradient	https://gradient.paperspace.com/
FloydHub	https://www.floydhub.com/
Kaggle.com	www.kaggle.com
easyaiforum.cn	https://easyaiforum.cn
Google Cloud Platform (GCP)	https://cloud.google.com/compute/
Azure	https://azure.microsoft.com/en-us/services/virtual-machines/data-science-virtual-machines/
Colab	https://colab.research.google.com/notebooks/welcome.ipynb
SageMaker	https://aws.amazon.com/sagemaker/
AWS EC2	https://aws.amazon.com/

Cost varies depending on CPU and GPU configurations
Example Gradient
CPU ~ $0.3/hour (Mid-range instance with 2 vCPU, 4GB RAM)
GPU ~ $0.51/hour (Low-cost instance with 8GB GPU, 8 vCPU, 30GB RAM)

Free with limitations (Kaggle.com)

Free with limitations (Colab)

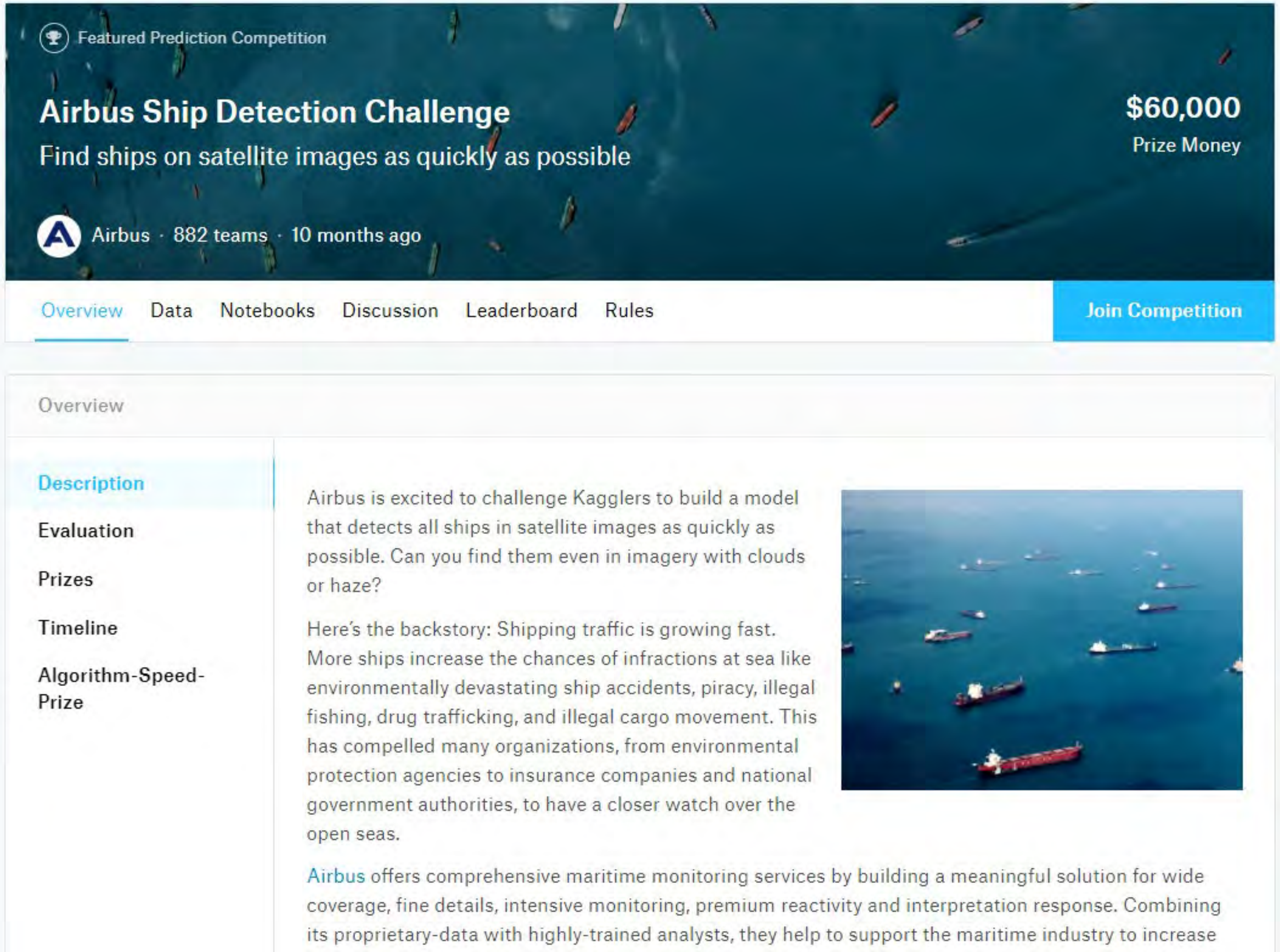
https://course.fast.ai/ Under server setup

Software

- Python/Anaconda environment
- Jupyter notebooks: “The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.”
- https://jupyter.org/
- Python libraries: fastai, sklearn, pandas, numpy, matplotlib, …

All free and
open source!!!

Example of How to Predict the 'Rare Event'



https://www.kaggle.com/c/airbus-ship-detection/overview

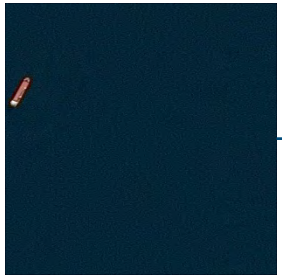
Main Ideas

- Computer vision and segmentation.
- Given an image, does it contain a ship?
- If the image contains a ship create a segmentation.
- Solution based on CNN (Convolutional Neural Networks) and Unet (encode/decode).
- ~200K training images (~26GB), ~ 35% contain ships. However, of the pictures that contain ships only a few pixels represent ships. A ship is a rare event!

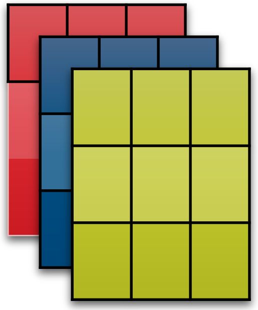
Creating a model that detects a ship

- The information of the pixels of an image is saved as an array with three channels: Red (R), Green (G), and Blue (B).

Input



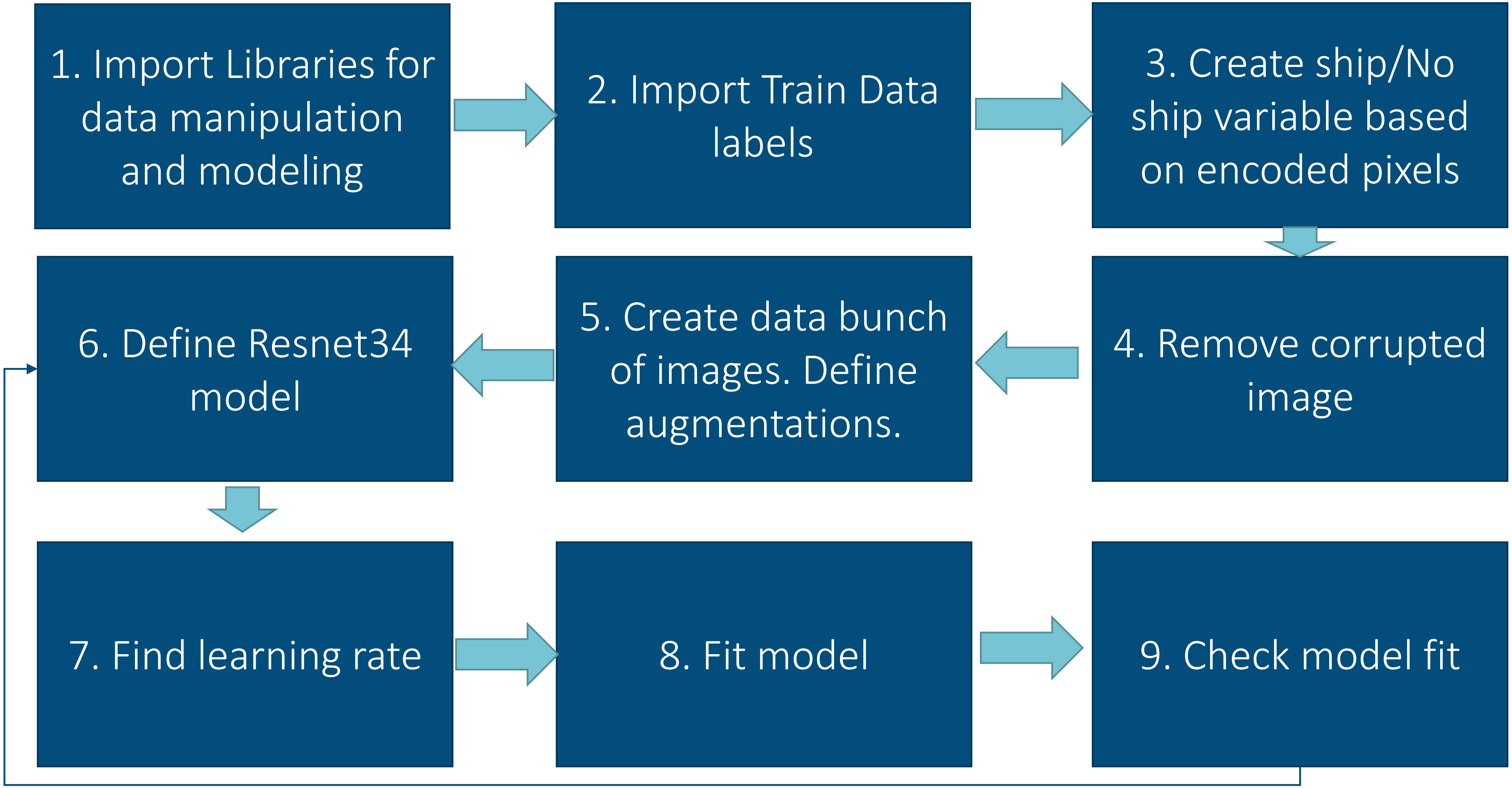
Represented as an array of RGB



Resnet 34 Convolutional Neural Network (CNN)

Prediction: Probability of Ship

Overview of the Solution



Python Code

```python
from fastai.vision import *
from fastai.metrics import error_rate
```

```python
bs = 64
#batch size: number of images that can be processed at a time. Depends on the GPU memory
# 1080Ti bs= 64
```

```python
#Path images
#Path annotations

path = 'E:/Kaggle/Airbus/'
path_img = path + 'data/train_v2/'
```

```python
import pandas as pd
data = pd.read_csv(path + "train_ship_segmentations_v2.csv")
```

20 lines of code ~97% accuracy detecting ships. ~30 minutes per epoch

```python
data['ship'] = data['EncodedPixels'].notnull()
data.drop(columns=['EncodedPixels'], inplace=True)
```

```python
#data cleaning: remove corrupted image
data.loc[data['ImageId'] == '6384c3e78.jpg']
#drop_imgs
data.drop([90158], inplace=True)
```

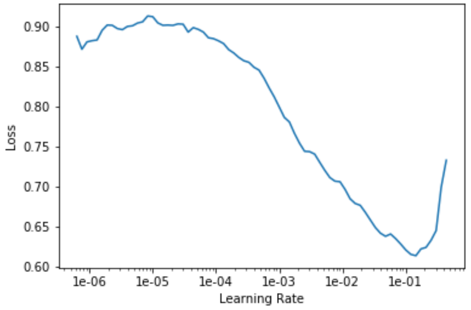
```python
#Data block to create data
data_train = ImageDataBunch.from_df(path_img, data, '', ds_tfms=get_transforms(),
                                    size=224, bs=bs).normalize(imagenet_stats)

#Note: in this particular dataset there was a "leakage" so some images overlap.
#A better solution needs to create
#a validation set without overlapping images.
```

```python
#define CNN - pretrained on imagenet
learn = cnn_learner(data_train, models.resnet34, metrics=accuracy)
```

```python
#find learning rate
learn.lr_find()
learn.recorder.plot()
```

```
LR Finder is complete, type {learner_name}.recorder.plot() to see the graph.
```

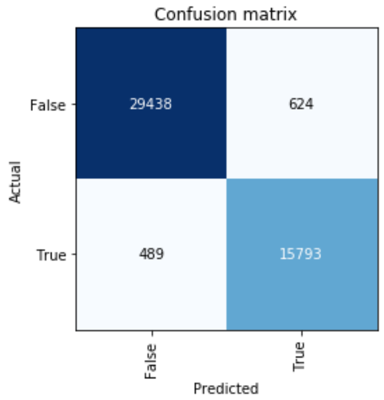



```python
#train
learn.fit_one_cycle(1, max_lr = 1e-3)
```

epoch	train_loss	valid_loss	accuracy	time
0	0.104120	0.069488	0.975984	27:23

```python
#interpret results
interp = ClassificationInterpretation.from_learner(learn)
```

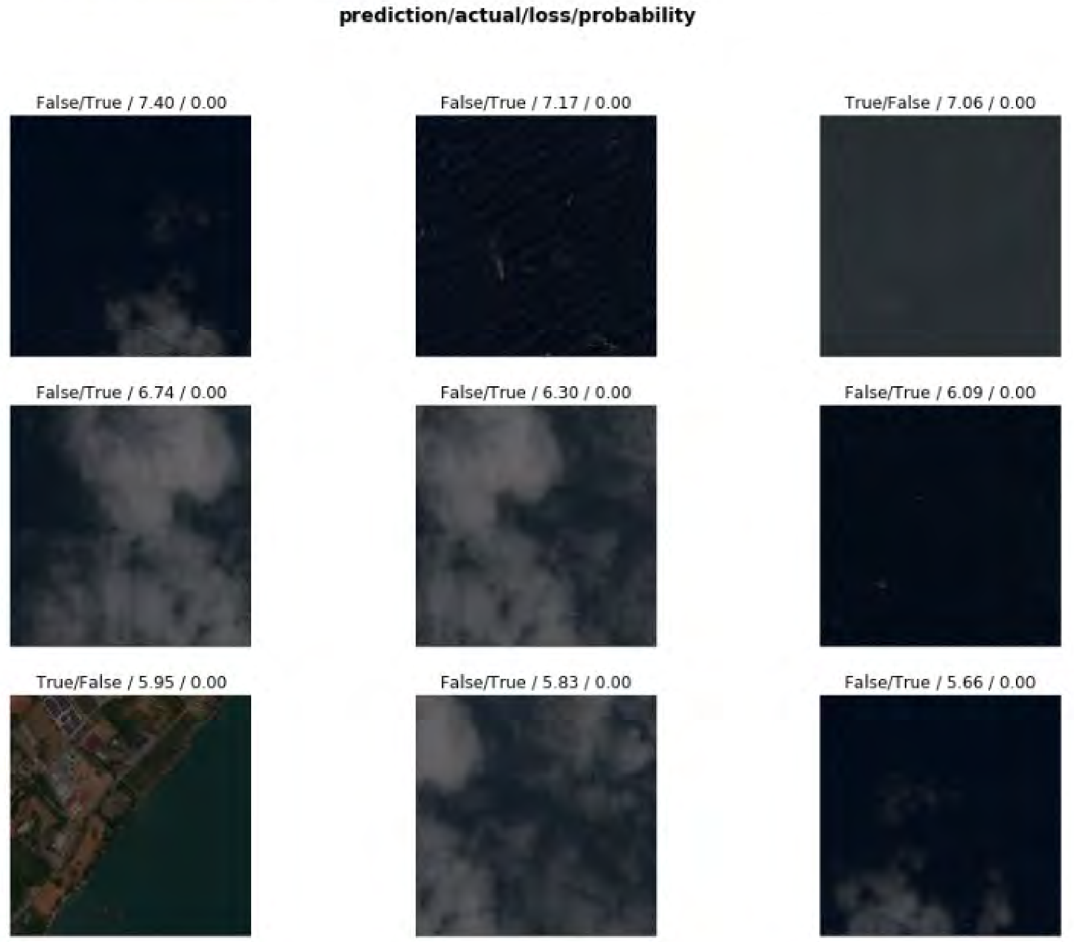
```python
interp.plot_confusion_matrix()
```



How to improve the model:

- Create a validation set without overlapping images.
- Train with increased image size.
- Unfreeze and train deeper layers.
- Predict using test time augmentation TTA.
- Use deeper networks such as resnet50.
- Blend of different models.

```
interp.plot_top_losses(9, figsize=(15,11))
```



Key Takeaways

- Kaggle is a good place for Actuaries to learn and practice predictive analytics skills.
- It is important to have the right hardware and software to be competitive in these competitions.
- Actuarial principles of data cleaning, bias/variance trade off, and detection of rare events are fundamental in machine learning.

Questions?

MMWellen@uams.edu