Applied Machine Learning
ML 101 &Course Introduction

James G. Shanahan [1,2]

[1]Church and Duncan Group,

[2]School of Informatics, Computing and Engineering, Indiana University

EMAIL: James_DOT_Shanahan_AT_gmail_DOT_com

Outline

- **AI/ML 101:**
  - Introduction
  - Linear Regression
  - Beyond linear regression
- **Top AI market trends** to watch in 2017 and beyond
  - Key technical developments
  - Case studies
- **ML investor or entrepreneur**
- **Short Case Study in a Python Notebook**
- **Conclusions**
- **Course Logistics**

MACHINE INTELLIGENCE 3.0

ENTERPRISE INTELLIGENCE

VISUAL
Orbital Insight, planet, clarifai, DEEPVISION, cortica, Iqocian, SPACE_KNOW, Captricity, netra, deepomatic

AUDIO
Gridspace, TalkIQ, nexidia, twilio, CAPIO, Expect Labs, Clover, Mobvoi, Qurious.AI, popUP archive

SENSOR
PREDIX, C3IoT, MAANA, Sentenai, PLANET OS, UPTAKE, IMUBIT, Preferred Networks, thingworx, KONUX, Alluvium

INTERNAL DATA
PRIMER, IBM WATSON, Cycorp, Palantir, ARIMO, Alation, sapho, Outlier, Digital Reasoning

MARKET
mattermark, Quid, DataFox, PREMISE, Bottlenose, MOTIVA, enigma, CBINSIGHTS, Tracxn, predata

ENTERPRISE FUNCTIONS

CUSTOMER SUPPORT
DigitalGenius, Kasisto, ELOQUENT, Wise.io, ACTIONIQ, zendesk, Preact, CLARABRIDGE

SALES
collective[i], 6sense, fusemachines, AVISO, salesforce, INSIDE SALES.COM, clari, Zensight

MARKETING
MINTIGO, Lattice, RADIUS, LiftIgniter, [PERSADO], brightfunnel, retention SCIENCE, COGNICOR, AIRPR, msg.ai

SECURITY
CYLANCE, DARKTRACE, ZIMPERIUM, deepinstinct, Sentinel, DEMISTO, graphistry, drawbridge, SignalSense, AppZen

RECRUITING
textio, entelo, Wade & Wendy, hiQ, unitive, SpringRole, GIGSTER, HireVue

AUTONOMOUS SYSTEMS

GROUND NAVIGATION
drive.ai, AdasWorks, ZOOX, MOBILEYE, UBER, Google, TESLA, nuTonomy, Auro Robotics

AERIAL
SKYDIO, SHIELD AI, Airware, dji, LILY, DroneDeploy, pilot.ai, SKYCATCH

INDUSTRIAL
JAYBRIDGE, OSARO, CLEARPATH ROBOTICS, fetch ROBOTICS, KINDRED, HARVEST AUTOMATION, rethink robotics

AGENTS

PERSONAL
amazon alexa, Cortana, Allo, facebook, Siri, Replika

PROFESSIONAL
butler.ai, pogo, SKIPFLAG, clara, x.ai, slack, talla, Zoom.ai, sudo

INDUSTRIES

AGRICULTURE
BLUE RIVER, mavrx, tule, TRACE GENOMICS, Pivot Bio, TerrAvion, AGRI-DATA, Descartes Labs, udio, abundant ROBOTICS

EDUCATION
KNEWTON, volley, gradescope, CTI, coursera, UDACITY, altschool

INVESTMENT
Bloomberg, sentient, iSENTIUM, KENSHO, alphasense, Dataminr, CEREBELLUM CAPITAL, Quandl

LEGAL
blueJ, BEAGLE, Everlaw, RAVEL, Seal, ROSS, LEGAL ROBOT

LOGISTICS
NAUTO, Acerta, PRETECKT, clearmetal, Routific, MARBLE, PITSTOP

INDUSTRIES CONT'D

MATERIALS
zymergen, Citrine, Eigen Innovations, SIGHT MACHINE, GINKGO BIOWORKS, nanotronics, CALCULARIO

RETAIL FINANCE
TALA, zestfinance, Lendo, earnest, affirm, MIRADOR, wealthfront, Betterment

HEALTHCARE

PATIENT
PULSE, CareSkore, ZEPHYR HEALTH, IBM Watson Health, Oncora, SENTRIAN, Atomwise, Numerate

IMAGE
BUTTERFLY, 3SCAN, ARTERYS, enlitic, BAYLABS, imagia, Google DeepMind

BIOLOGICAL
iCarbonX, color, GRAIL, deep genomics, RECURSION, LUMINIST, Numerate, Atomwise, verily, WHOLE BIOME

shivonzilis.com/MACHINEINTELLIGENCE · Bloomberg BETA

TECHNOLOGY STACK

AGENT ENABLERS
OCTANE.AI, howdy, MaluubA, KITT.AI, OpenAI Gym, Kasisto, AUTOMAT, semanticmachines

DATA SCIENCE
DOMINO, SPARKBEYOND, rapidminer, kaggle, DataRobot, yhat, AYASDI, data iku, seldon, yseop, bigml

MACHINE LEARNING
CognitiveScale, GoogleML, context relevant, Cycorp, HyperScience, narralogics, minds.ai, H2O.ai, SCALED INFERENCE, sparkcognition, loop, GEOMETRIC INTELLIGENCE, deepsense.io, reactive, skymind, bonsai

NATURAL LANGUAGE
agolo, AYLIEN, LEXALYTICS, Narrative Science, loop.ai Labs, spaCy, LUMINOSO, cortical.io, MonkeyLearn

DEVELOPMENT
SIGOPT, HyperOpt, fuzzyio, pkite, rainforest, lobe, Anodot, Signifai, LAYER 6 AI, bonsai

DATA CAPTURE
CrowdFlower, diffbot, CrowdAI, import.io, Paxata, DATASIFT, amazon mechanical turk, enigma, WorkFusion, DATALOGUE, TRIFACTA, parsehub

OPEN SOURCE LIBRARIES
Keras, Chainer, CNTK, TensorFlow, Caffe, H20, DEEPLEARNING4J, theano, torch, DSSTNE, Scikit-learn, AzureML, neon, MXNet, DMTK, Spark, PaddlePaddle, WEKA

HARDWARE
KNUPATH, TENSTORRENT, Cirrascale, NVIDIA, intel nervana, Movidius, tensilica, GoogleTPU, 10^{26} Labs, Qualcomm, Cerebras, Isosemi

RESEARCH
OpenAI, mnaisense, ELEMENT AI, vicarious, KNOGGIN, Numenta, Kimera Systems, Cogitai

AI Revenue and M&As

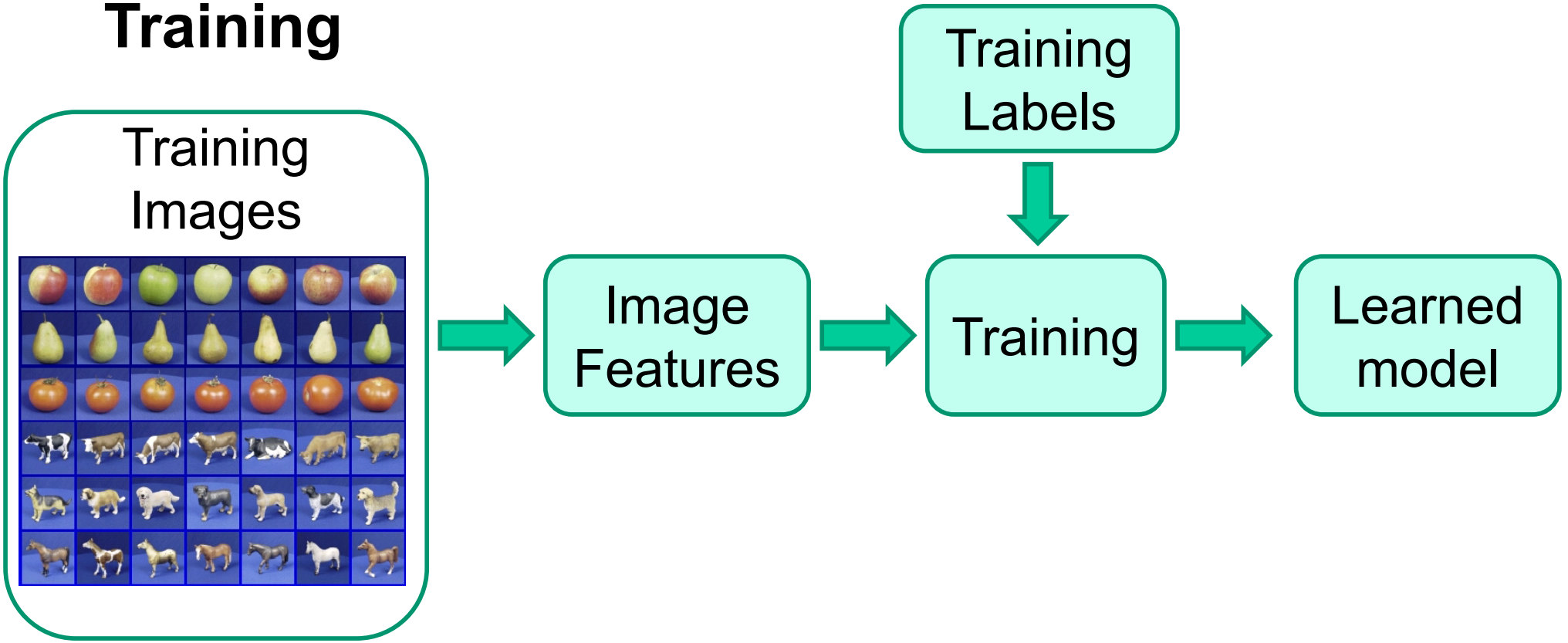
- **AI Software: from $1.38 billion in 2016 to $59.75 billion by 2025 [Tractica.com 2017]**
  - Autonomous vehicles, Consumer/social/image, Stock trading, Healthcare, HR, patent, security, digitizing paper
  - 9-years with a compound annual growth rate (CAGR) of 52%.
- **AI Services: $120 Billion by 2025**
  - training, integration, Apps
- **AI Hardware: $120 Billion by 2025**
  - GPUs, CPUs, Customer
- **250 AI acquisitions 2012-2017**
  - Over 250 private companies using AI algorithms across different verticals have been acquired 2012-2017(Q1), with 37 acquisitions taking place in Q1 2017 alone. Google acq 12 companies.

Outline

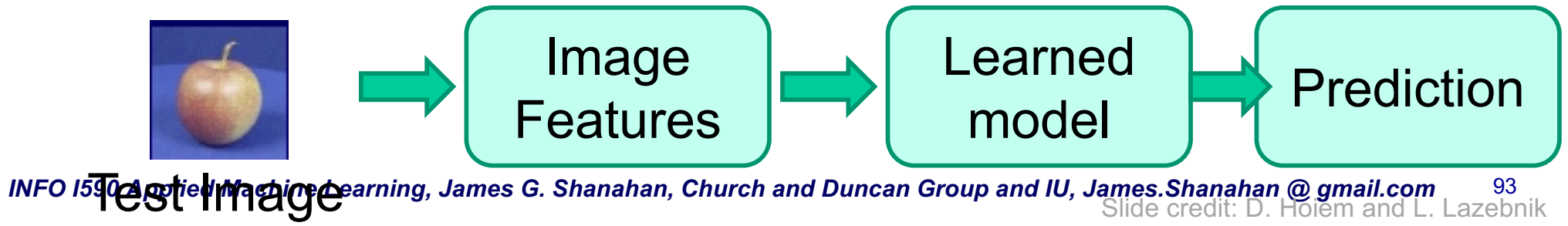
- **AI/ML 101:**
  - Introduction
  - Linear Regression
  - Beyond linear regression
- **Top AI market trends** to watch in 2017 and beyond
  - Key technical developments
  - Case studies
- **ML investor or entrepreneur**
- **Short Case Study in a Python Notebook**
- **Conclusions**
- **Course Logistics**

Steps

Training



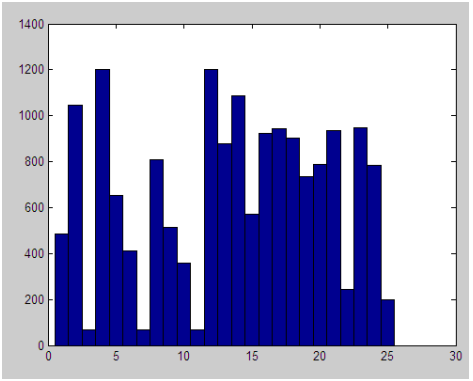
Testing

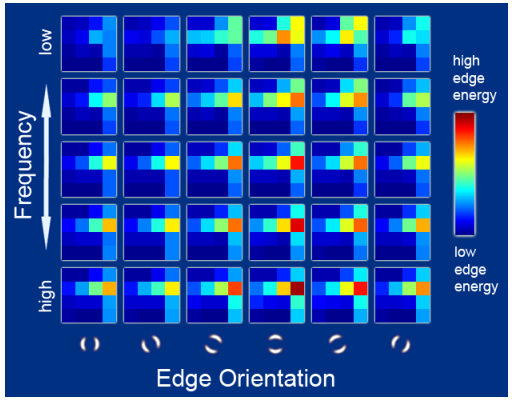


Slide credit: D. Hoiem and L. Lazebnik

Features

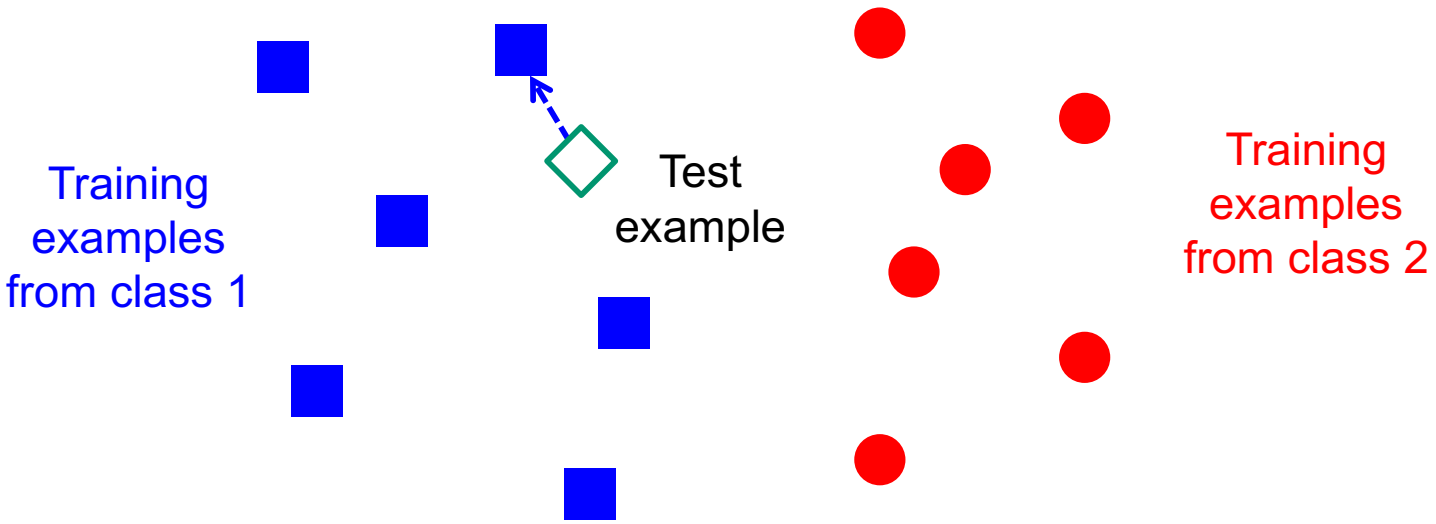
- **Raw pixels**
- **Histograms**
- **GIST descriptors**
- **…**





Slide credit: L. Lazebnik

Classifiers: Nearest neighbor

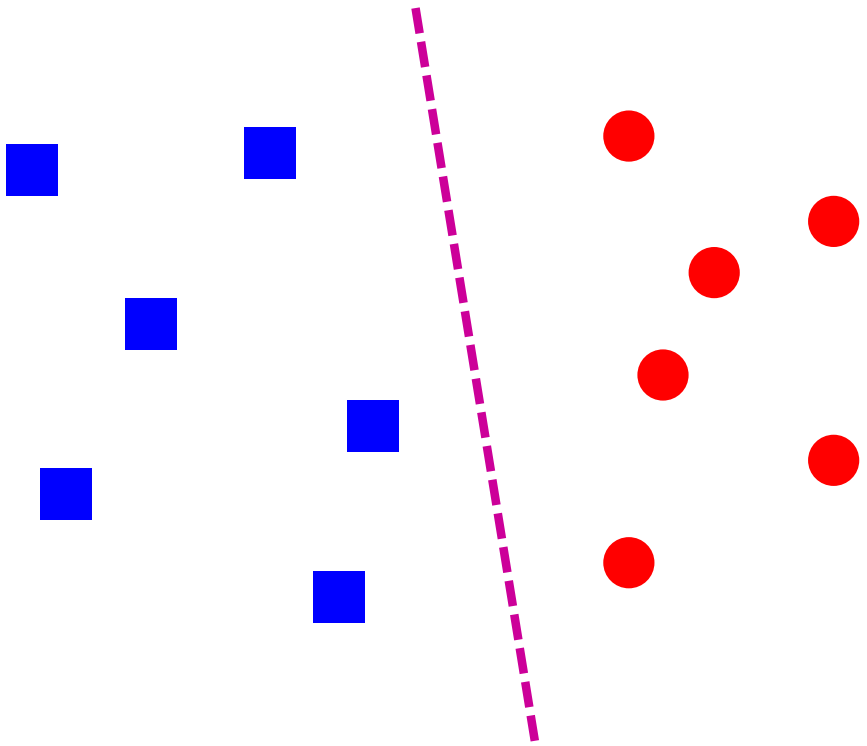


f(x) = label of the training example nearest to x

- **All we need is a distance function for our inputs**
- **No training required!**

Slide credit: L. Lazebnik

Classifiers: Linear



- Find a *linear function* to separate the classes:

$$f(x) = \text{sgn}(w \cdot x + b)$$

Slide credit: L. Lazebnik

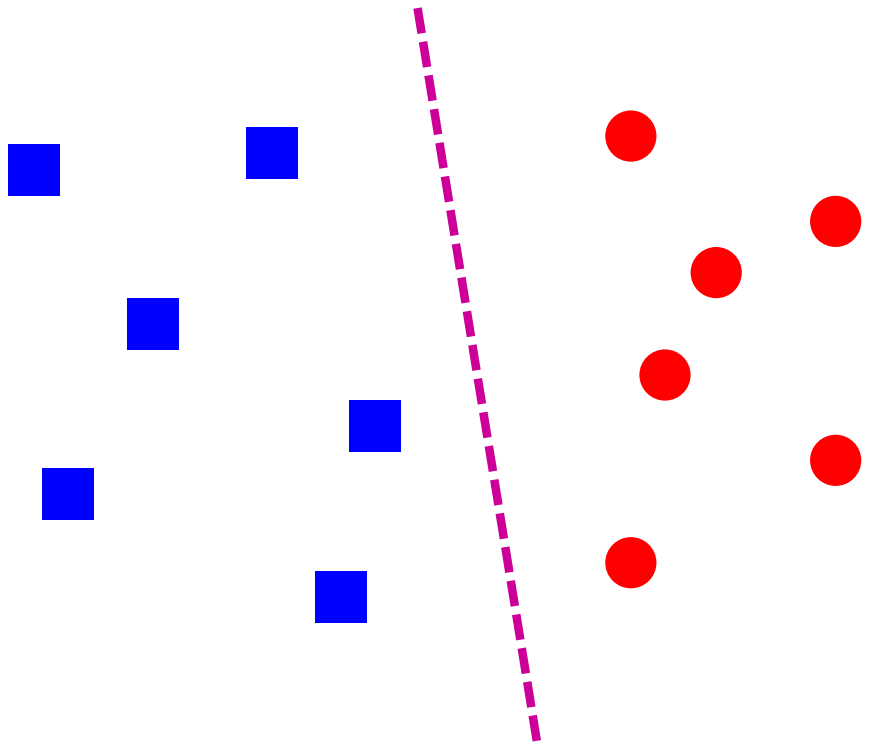
Many classifiers to choose from

- **SVM**
- **Neural networks**
- **Naïve Bayes**
- **Bayesian network**
- **Logistic regression**
- **Randomized Forests**
- **Boosted Decision Trees**
- **K-nearest neighbor**
- **RBMs**
- **Etc.**

Which is the best one?

Slide credit: D. Hoiem

Classifiers: Linear



- Find a *linear function* to separate the classes:

$$f(x) = \text{sgn}(w \cdot x + b)$$

Slide credit: L. Lazebnik

Comparison

assuming x in {0 1}

	Learning Objective	Training	Inference
Naïve Bayes	$\text{maximize} \sum_i \left[\sum_j \log P(x_{ij} \mid y_i; \theta_j) + \log P(y_i; \theta_0) \right]$	$\theta_{kj} = \frac{\sum_i \delta(x_{ij} = 1 \wedge y_i = k) + r}{\sum_i \delta(y_i = k) + Kr}$	$\theta_1{}^T \mathbf{x} + \theta_0{}^T (1 - \mathbf{x}) > 0$ where $\theta_{1j} = \log \frac{P(x_j = 1 \mid y = 1)}{P(x_j = 1 \mid y = 0)}$, $\theta_{0j} = \log \frac{P(x_j = 0 \mid y = 1)}{P(x_j = 0 \mid y = 0)}$
Logistic Regression	$\text{maximize} \sum_i \log(P(y_i \mid \mathbf{x}, \boldsymbol{\theta})) + \lambda \lVert \boldsymbol{\theta} \rVert$ where $P(y_i \mid \mathbf{x}, \boldsymbol{\theta}) = 1/(1 + \exp(-y_i \boldsymbol{\theta}^T \mathbf{x}))$	Gradient ascent	$\boldsymbol{\theta}^T \mathbf{x} > 0$
Linear SVM	$\text{minimize } \lambda \sum_i \xi_i + \frac{1}{2} \lVert \boldsymbol{\theta} \rVert$ such that $y_i \boldsymbol{\theta}^T \mathbf{x} \geq 1 - \xi_i \quad \forall i$	Linear programming	$\boldsymbol{\theta}^T \mathbf{x} > 0$
Kernelized SVM	complicated to write	Quadratic programming	$\sum_i y_i \alpha_i K(\hat{\mathbf{x}}_i, \mathbf{x}) > 0$
Nearest Neighbor	most similar features → same label	Record data	y_i where $i = \underset{i}{\text{argmin}} \; K(\hat{\mathbf{x}}_i, \mathbf{x})$

Slide credit: D. Hoiem

Titanic: predict passenger survival

- **Task: Predict whether a passenger will survive or not**
  - you must predict the fate of the passengers aboard the RMS Titanic, which famously sank in the Atlantic ocean during its maiden voyage from the UK to New York City after colliding with an iceberg.

Titanic: Python and R

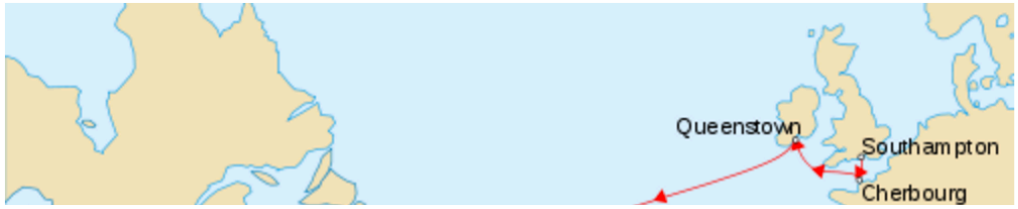
- **In python:**
  - http://nbviewer.jupyter.org/urls/dl.dropbox.com/s/zshj6zmo8q6ej85/Titanic-EDA-LogisticRegression.ipynb
- **In R**
  - http://trevorstephens.com/kaggle-titanic-tutorial/getting-started-with-r/

Titanic: Getting Started With R

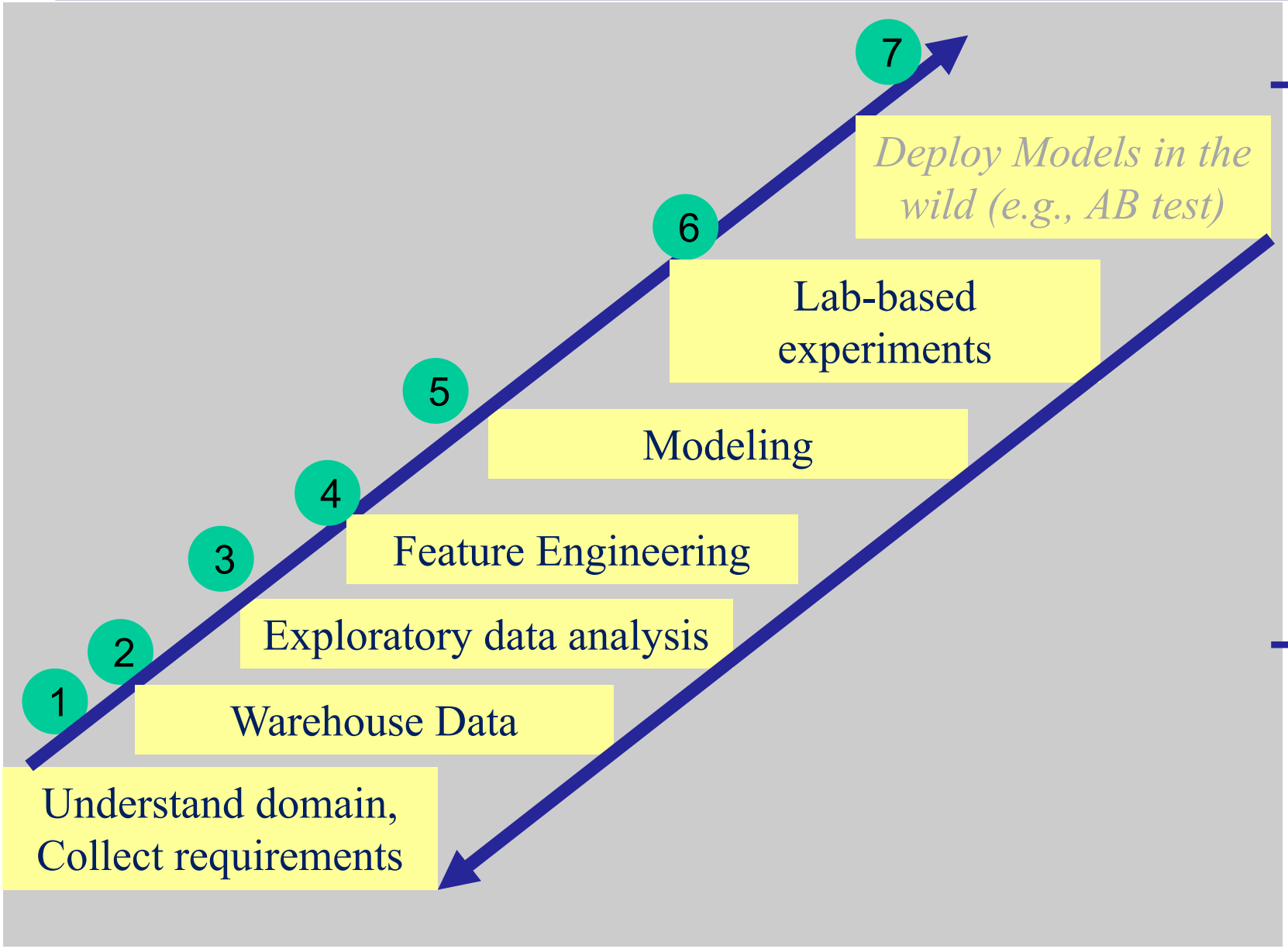
3 minutes read

So you're excited to get into prediction and like the look of Kaggle's excellent getting started competition, Titanic: Machine Learning from Disaster? Great! It's a wonderful entry-point to machine learning with a manageably small but very interesting dataset with easily understood variables.

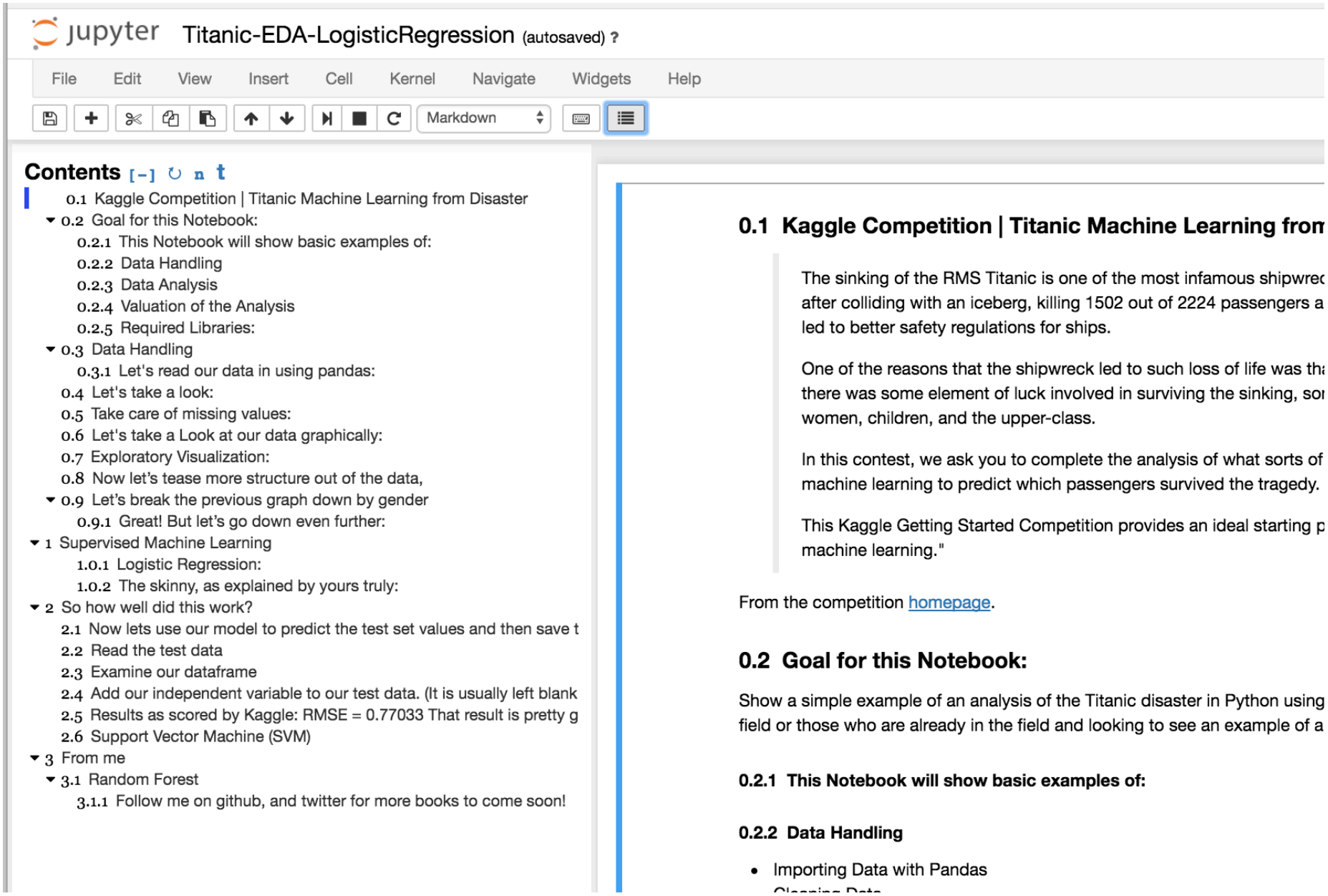
In this competition, you must predict the fate of the passengers aboard the RMS Titanic, which famously sank in the Atlantic ocean during its maiden voyage from the UK to New York City after colliding with an iceberg.



Typical Abstract Data Analytics Pipeline



Python Notebook



Python Notebook

Goal for this Notebook:

Show a simple example of an analysis of the Titanic disaster in Python using a full complement of PyData utilities. This is aimed for those looking to get into the field or those who are already in the field and looking to see an example of an analysis done with Python.

This Notebook will show basic examples of:

Data Handling

- Importing Data with Pandas
- Cleaning Data
- Exploring Data through Visualizations with Matplotlib

Data Analysis

- Supervised Machine learning Techniques:
  - Logit Regression Model
  - Plotting results
  - Support Vector Machine (SVM) using 3 kernels
  - Basic Random Forest
  - Plotting results

Valuation of the Analysis

- K-folds cross validation to valuate results locally
- Output the results from the IPython Notebook to Kaggle

Read in the data as a dataframe

0.3 Data Handling

0.3.1 Let's read our data in using pandas:

```
]: df = pd.read_csv("data/train.csv")
```

Show an overview of our data:

```
]: df
```

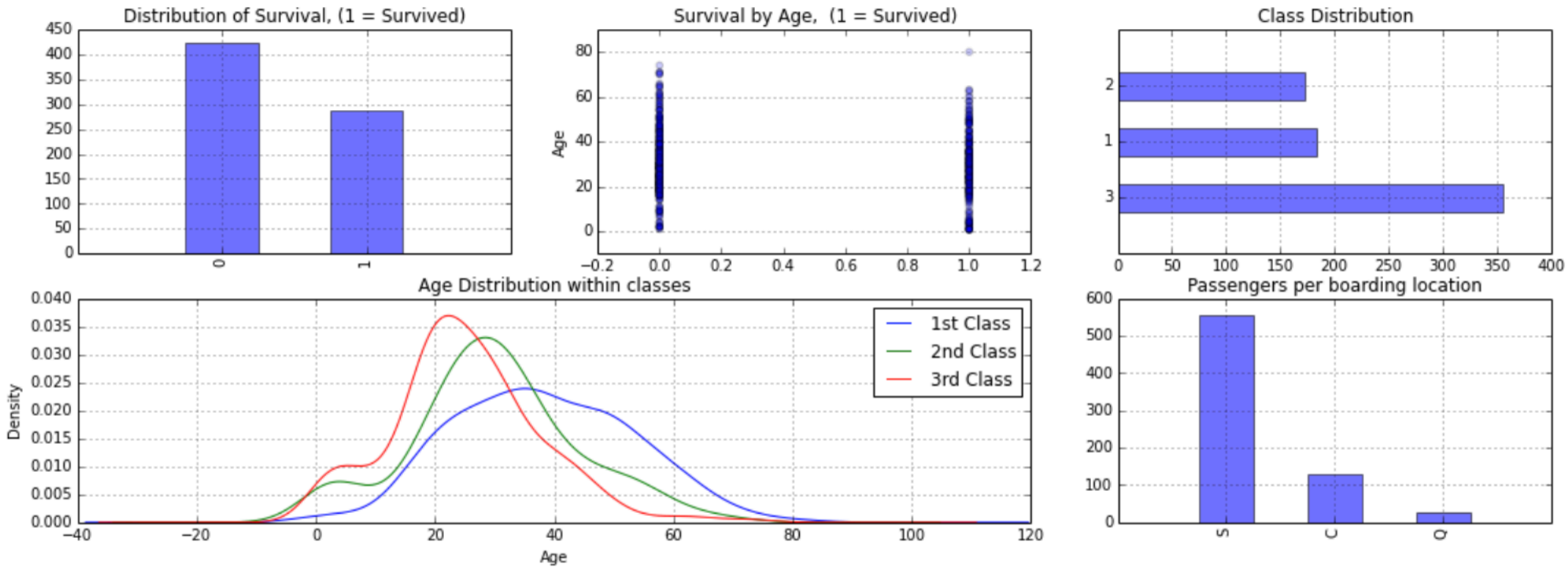
]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909	21.0750	NaN	S

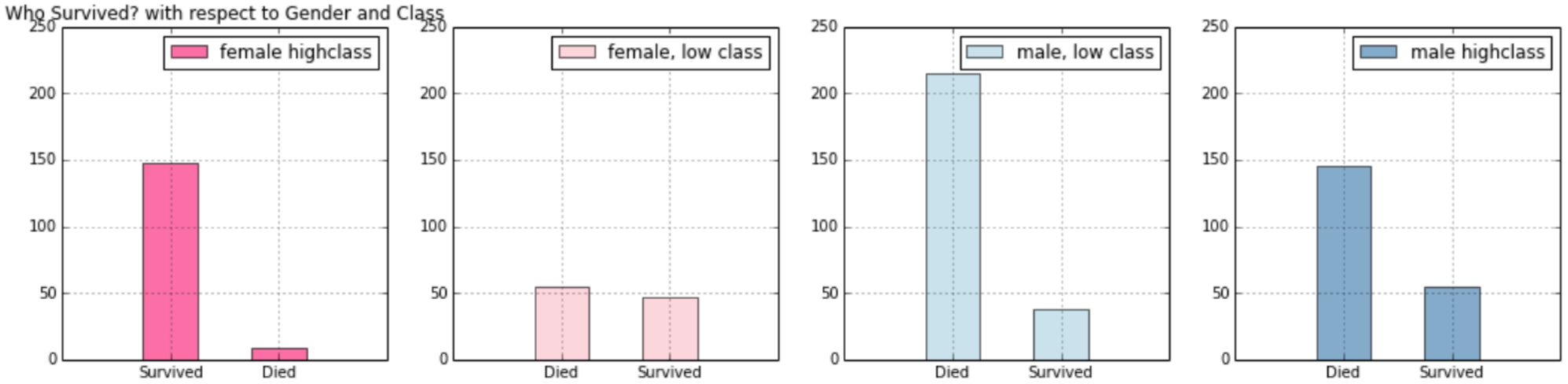
Exploratory Data Analysis

```
# specifies the parameters of our graphs
plt.title("Passengers per boarding location")
```

```
<matplotlib.text.Text at 0x119c1dd90>
```



Exploratory Data Analysis



Model passenger survival
Classification task using logistic regression

```
import statsmodels.api as sm
```

```
# model formula
# here the ~ sign is an = sign, and the features of our dataset
# are written as a formula to predict survived. The C() lets our
# regression know that those variables are categorical.
# Ref: http://patsy.readthedocs.org/en/latest/formulas.html
formula = 'Survived ~ C(Pclass) + C(Sex) + Age + SibSp  + C(Embarked)'
# create a results dictionary to hold our regression results for easy analysis later
results = {}
```

```
# create a regression friendly dataframe using patsy's dmatrices function
y,x = dmatrices(formula, data=df, return_type='dataframe')

# instantiate our model
model = sm.Logit(y,x)

# fit our model to the training data
res = model.fit()

# save the result for outputing predictions later
results['Logit'] = [res, formula]
res.summary()
```

Ensemble of trees

```
|:  # import the machine learning library that holds the randomforest
    import sklearn.ensemble as ske

    # Create the random forest model and fit the model to our training data
    y, x = dmatrices(formula_ml, data=df, return_type='dataframe')
    # RandomForestClassifier expects a 1 demensional NumPy array, so we convert
    y = np.asarray(y).ravel()
    #instantiate and fit our model
    results_rf = ske.RandomForestClassifier(n_estimators=100).fit(x, y)

    # Score the results
    score = results_rf.score(x, y)
    print "Mean accuracy of Random Forest Predictions on the data was: {0}".format(score)
```

R Code for Titanic

http://trevorstephens.com/kaggle-titanic-tutorial/getting-started-with-r/

- **All code is available on myGithub repository.**
- **A series of tutorials in five parts is available here:**
  - Part 1: Booting Up R
  - Part 2: The Gender-Class Model
  - Part 3: Decision Trees
  - Part 4: Feature Engineering
  - Part 5: Random Forests

New features: Title

- **Title feature (extracted from Name)**

```
> combi$Name <- as.character(combi$Name)
> combi$Name[1]
[1] "Braund, Mr. Owen Harris"
```

```
> table(combi$Title)
       Capt         Col         Don        Dona          Dr    Jonkheer        Lady
          1           4           1           1           8           1           1
      Major      Master        Miss        Mlle         Mme          Mr         Mrs
          2          61         260           2           1         757         197
         Ms         Rev         Sir         the Countess
          2           8           1           1
```

Reduce titles to 2 groups

For the ladies, we have Dona, Lady, Jonkheer (*see comments below), and of course our Countess. All of these are again the rich folks, and may have acted somewhat similarly due to their noble birth. Let's combine these two groups and reduce the number of factor levels to something that a decision tree might make sense of:

```
> combi$Title[combi$Title %in% c('Capt', 'Don', 'Major', 'Sir')] <- 'Sir'
< ncombi$Title[combi$Title %in% c('Dona', 'Lady', 'the Countess', 'Jonkheer')] <- 'Lady'
```

Our final step is to change the variable type back to a factor, as these are essentially categories that we have created:

```
> combi$Title <- factor(combi$Title)
```

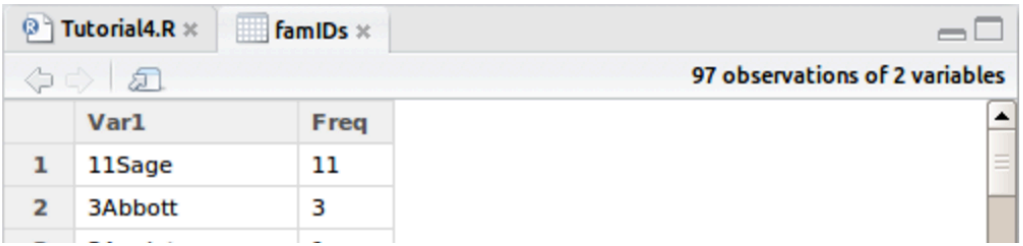
Family variable

```
> table(combi$FamilyID)
          11Sage          3Abbott        3Appleton        3Beckwith          3Boulos
              11                3                1                2                3
        3Bourke          3Brown        3Caldwell        3Christy        3Collyer
               3                4                3                2                3
        3Compton         3Cornell         3Coutts         3Crosby         3Danbom
               3                1                3                3                3 . . .
```

Hmm, a few seemed to have slipped through the cracks here. There's plenty of FamilyIDs with only one or two members, even though we wanted only family sizes of 3 or more. Perhaps some families had different last names, but whatever the case, all these one or two people groups is what we sought to avoid with the three person cut-off. Let's begin to clean this up:

```
> famIDs <- data.frame(table(combi$FamilyID))
```

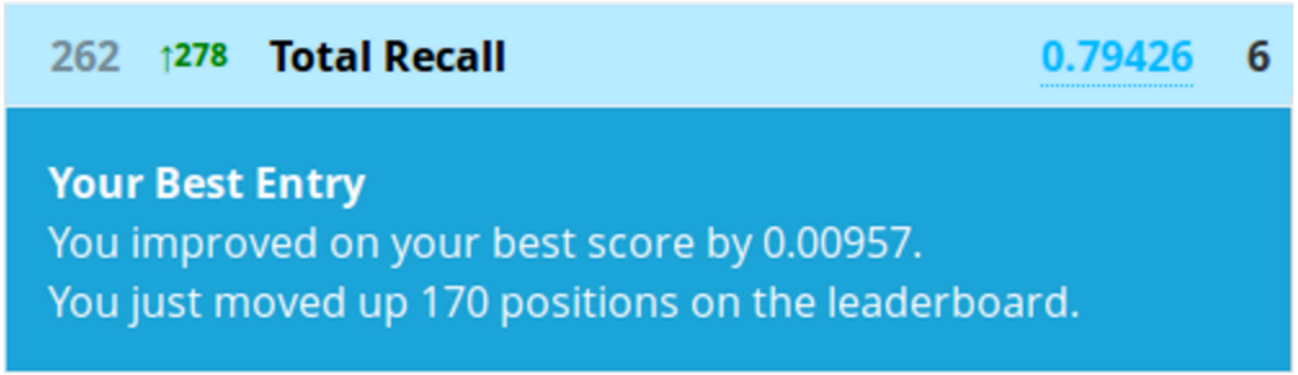
Now we have stored the table above to a dataframe. Yep, you can store most tables to a dataframe if you want to, so let's take a look at it by clicking on it in the explorer:



Improved score with new features + DT

- ..

But all that aside, you know should know how to create a submission from a decision tree, so let's see how it performed!



Exercise

- **Download Titanic sample code**
- **Do a submission and see where you come on the Kaggle dashboard**

https://www.kaggle.com/c/titanic

Knowledge • 4,775 teams

Titanic: Machine Learning from Disaster

Fri 28 Sep 2012 — Sat 31 Dec 2016 (3 months to go)

Dashboard

- Home
  - Data
  - Make a submission
- Information
  - Description
  - Evaluation
  - Rules
  - Prizes
  - Frequently Asked Questio...
  - Getting Started With Excel
  - Getting Started With Pyth...
  - Getting Started With Pyth...
  - Getting Started With Rand...
  - New: Getting Started with R
  - Submission Instructions
- Forum
- Kernels
  - New Script
  - New Notebook
- Leaderboard
- Visualization
- My Submissions

Competition Details » Get the Data » Make a submission

Predict survival on the Titanic using Excel, Python, R & Random Forests

If you're new to data science and machine learning, or looking for a simple intro to the Kaggle competitions platform, this is the best place to start. Continue reading below the competition description to discover a number of tutorials, benchmark models, and more.

Competition Description

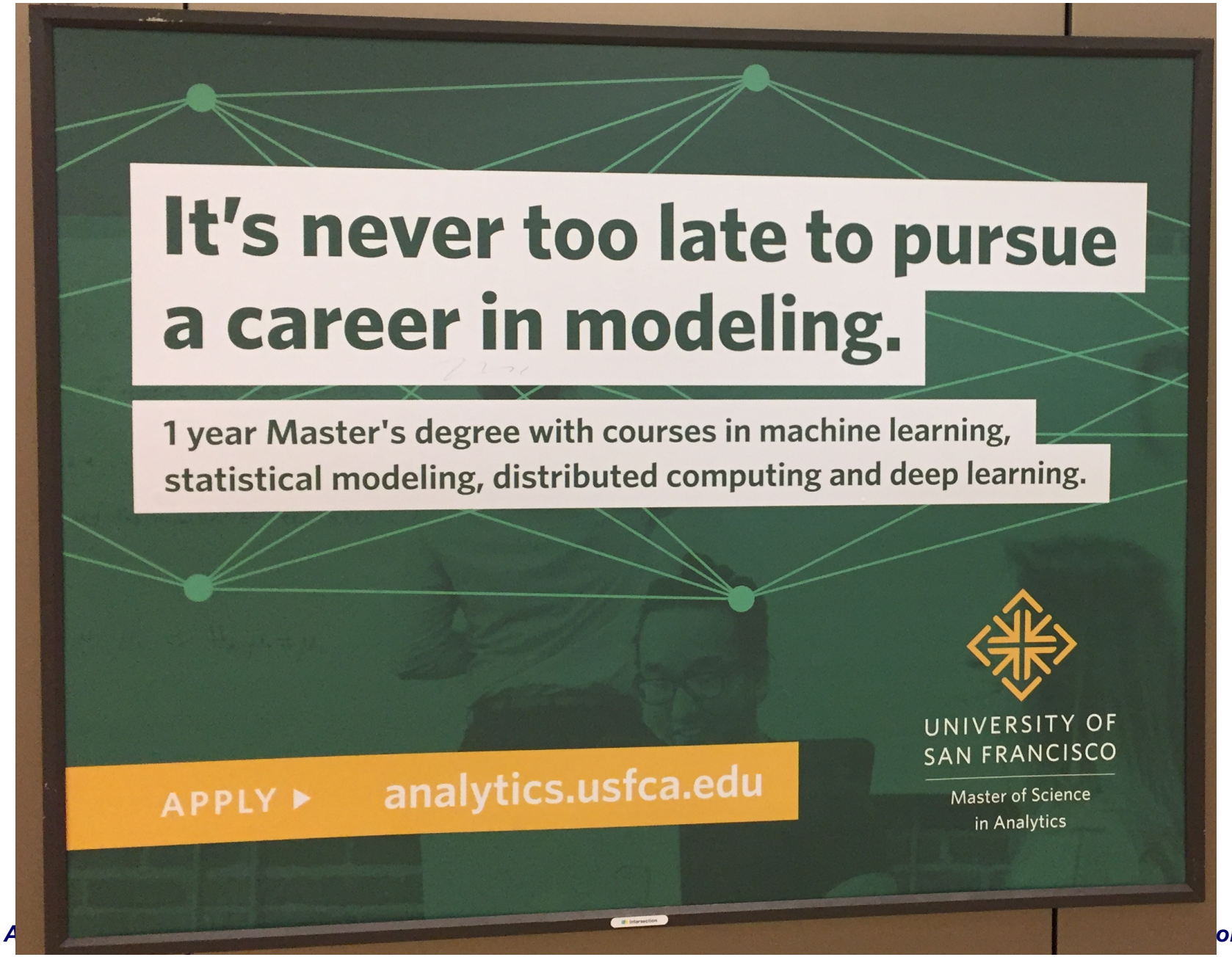
The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

Outline

- **AI/ML 101:**
  - Introduction
  - Linear Regression
  - Beyond linear regression
- **Top AI market trends** to watch in 2017 and beyond
  - Key technical developments
  - Case studies
- **ML investor or entrepreneur**
- **Short Case Study in a Python Notebook**
- **Conclusions**
- **Course Logistics**

On The BART train in San Francisco



9/2017

Conclusions

Machine learning will change how we work, rest and play

- 20th century life transformers: Electricity, automobile, Internet, mobile phones **Analytical→ Predictive → Decisive**
- Executives are realizing that this new technology, AI, could change everything, but nobody knows exactly how or when
- Companies have at their disposal the full set of building blocks to begin embedding machine intelligence in their businesses.

Education:

- Tools; teach machine learning using automatic differentiation

Social impact:

- Creating a workforce that combines lower headcount with higher productivity
- Ethical concerns

Huge opportunities

- AI: USD 6.5 billion (WorldWide) in 2016 → $300B by 2025
- Corporate Arms Race for AI startups (250 acqs in last 5 years)

Outline

- **AI/ML 101:**
  - Introduction
  - Linear Regression
  - Beyond linear regression
- **Top AI market trends** to watch in 2017 and beyond
  - Key technical developments
  - Case studies
- **ML investor or entrepreneur**
- **Short Case Study in a Python Notebook**
- **Conclusions**
- **Course Logistics**

Course Schedule

PART 1: Introduction

1. Machine Learning introduction and course overview
2. KNN: classification, regression + EDA + ML pipelines
3. Optimization theory: gradient descent

PART 2: Supervised machine learning

4. Linear Regression
5. Probabilistic approaches to ML
6. Classification: logistic/softmax regression
7. **MID-TERM exam**
8. Perceptron and Support Vector Machines
9. Non-gradient-based approaches: decision trees, random forest

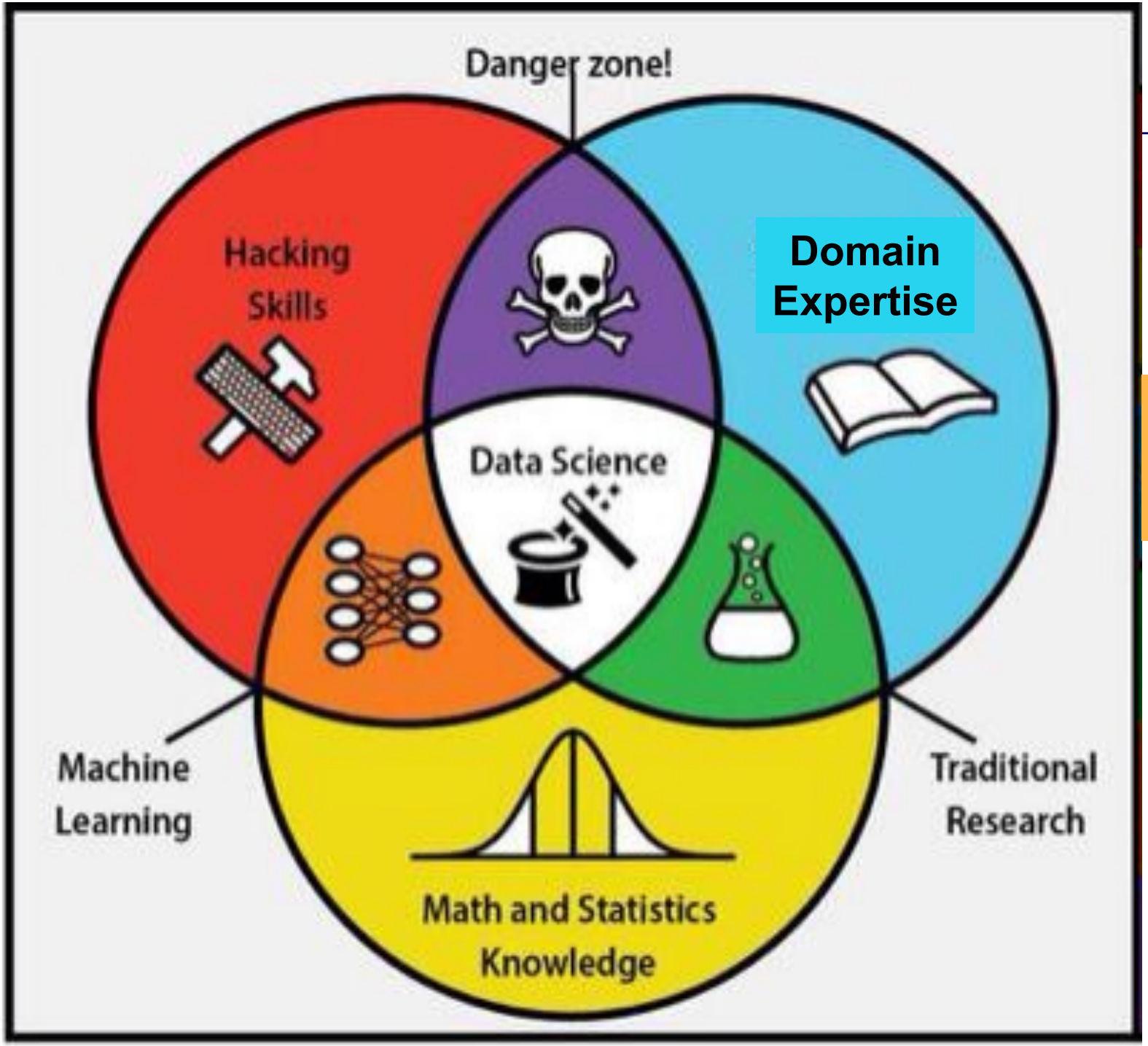
PART 3: Unsupervised learning

10. Clustering: K-means, Hierarchical clustering, DBSCAN
11. Dimensionality reduction: PCA, tSNE, Word2Vec

PART 4: Advanced/Recent topics

12. Neural networks: backprop, core concepts, deep learning
13. Recommender systems: popularity-based, item-to-item, matrix factorizations
14. Reinforcement learning: multi-armed bandit, Bellman equation, Q-learning
15. **FINAL Exam**

DS Skillset



A venn diagram with a Danger Bearing

[Drew Conway]

Implementing algorithms from scratch

- **Focus on theory, implementation and practice**
- **Avoid the "Danger Zone" (lots of great libraries but….)**
- **There are several different reasons why implementing algorithms from scratch can be useful:**
  - it can help us to understand the inner workings of an algorithm
  - Let's us figure how to parallize
  - we can add new features to an algorithm or experiment with different variations of the core idea
  - we want to invent new algorithms or implement algorithms no one has implemented/shared yet
  - we are not satisfied with the API and/or we want to integrate it more "naturally" into an existing software library
  - we could try to implement an algorithm more efficiently
  - we circumvent licensing issues (e.g., Linux vs. Unix) or platform restrictions

http://cs231n.github.io/python-numpy-tutorial/

Python Numpy Tutorial

```
distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
min_index = np.argmin(distances) # get the index with smallest distance
Ypred[i] = self.ytr[min_index] # predict the label of the nearest example
```

Grading Scale

Grading Scale

Activity	Score
Assignments	20%
Midterm exam	30%
Class participation	20%
End of semester exam	30%

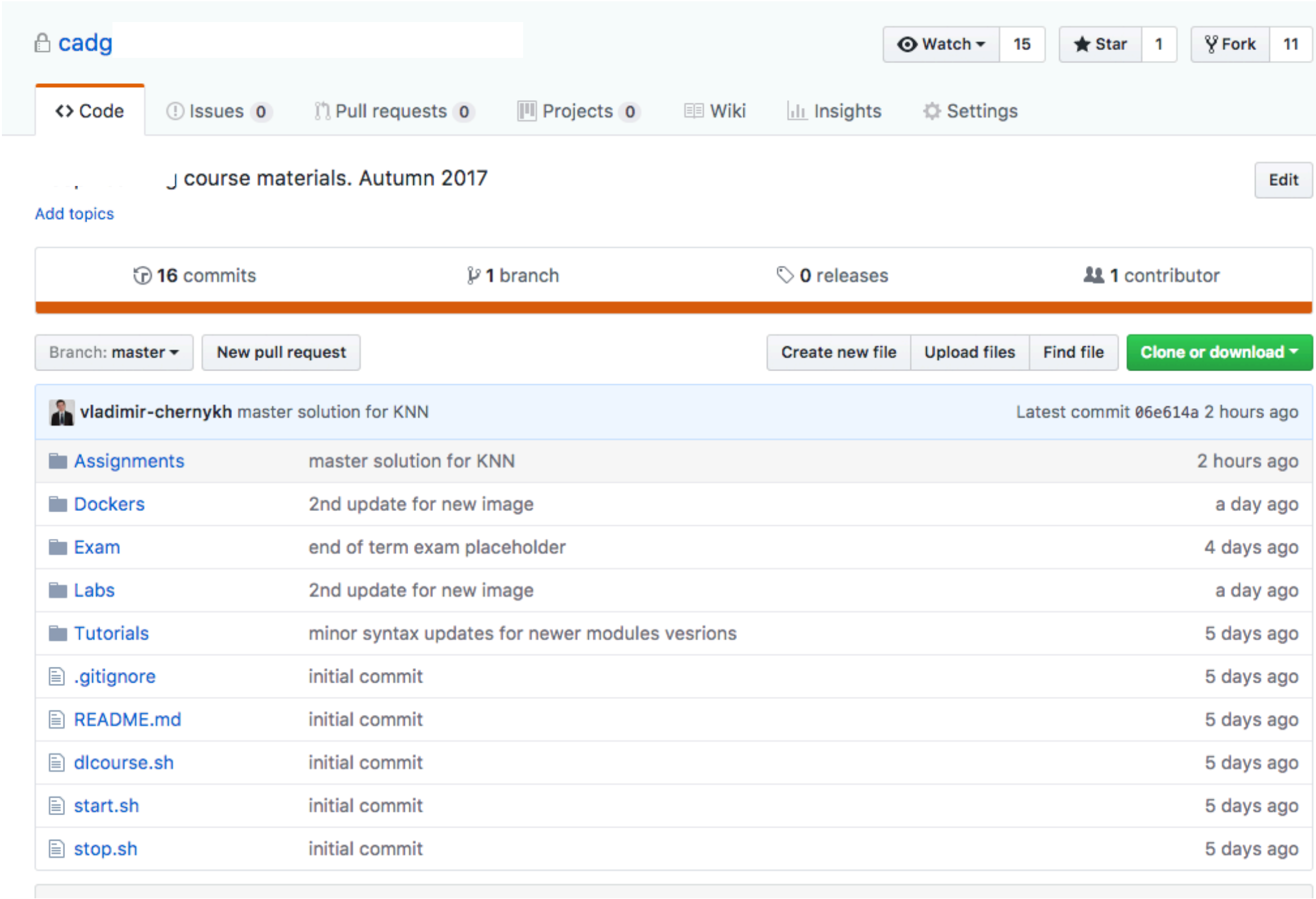
Final Grade	Total Score
A “Excellent”	>= 85%
B “Good”	from 75% to 84%
C “Satisfactory”	from 60% to 74%
D “Poor”	from 50% to 59%
F “Unacceptable”	< 50%

Weekly Schedule

- **One Practical session: to be announced**
- **Office hours on demand: to be announced**
- **Homework assignments: one every 2 weeks**
- **Use the STAR methodology to present you Questions/Answers**
  - Knowledge base questions
  - Problem solving

Course Resources

- **LMS: Canvas**
- **Github**
- **Docker Container**


cadg
Watch 15
Star 1
Fork 11
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
course materials. Autumn 2017
Edit
Add topics
16 commits
1 branch
0 releases
1 contributor
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
vladimir-chernykh master solution for KNN
Latest commit 06e614a 2 hours ago
Assignments master solution for KNN 2 hours ago
Dockers 2nd update for new image a day ago
Exam end of term exam placeholder 4 days ago
Labs 2nd update for new image a day ago
Tutorials minor syntax updates for newer modules vesrions 5 days ago
.gitignore initial commit 5 days ago
README.md initial commit 5 days ago
dlcourse.sh initial commit 5 days ago
start.sh initial commit 5 days ago
stop.sh initial commit 5 days ago

This class will be demanding

But it is a high RoI class

- **Plan on spending 10 hours +/-10 hours per week on this class**

End of lecture