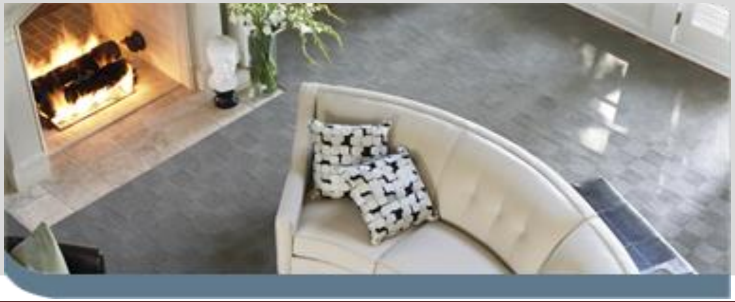
MHK Claims AI Scoring Predictive Analytics
- Update

Xiaomo Jiang
Yatkwai Kee
Richard Yan
Deepak Jhamta
Jithendra Koduru

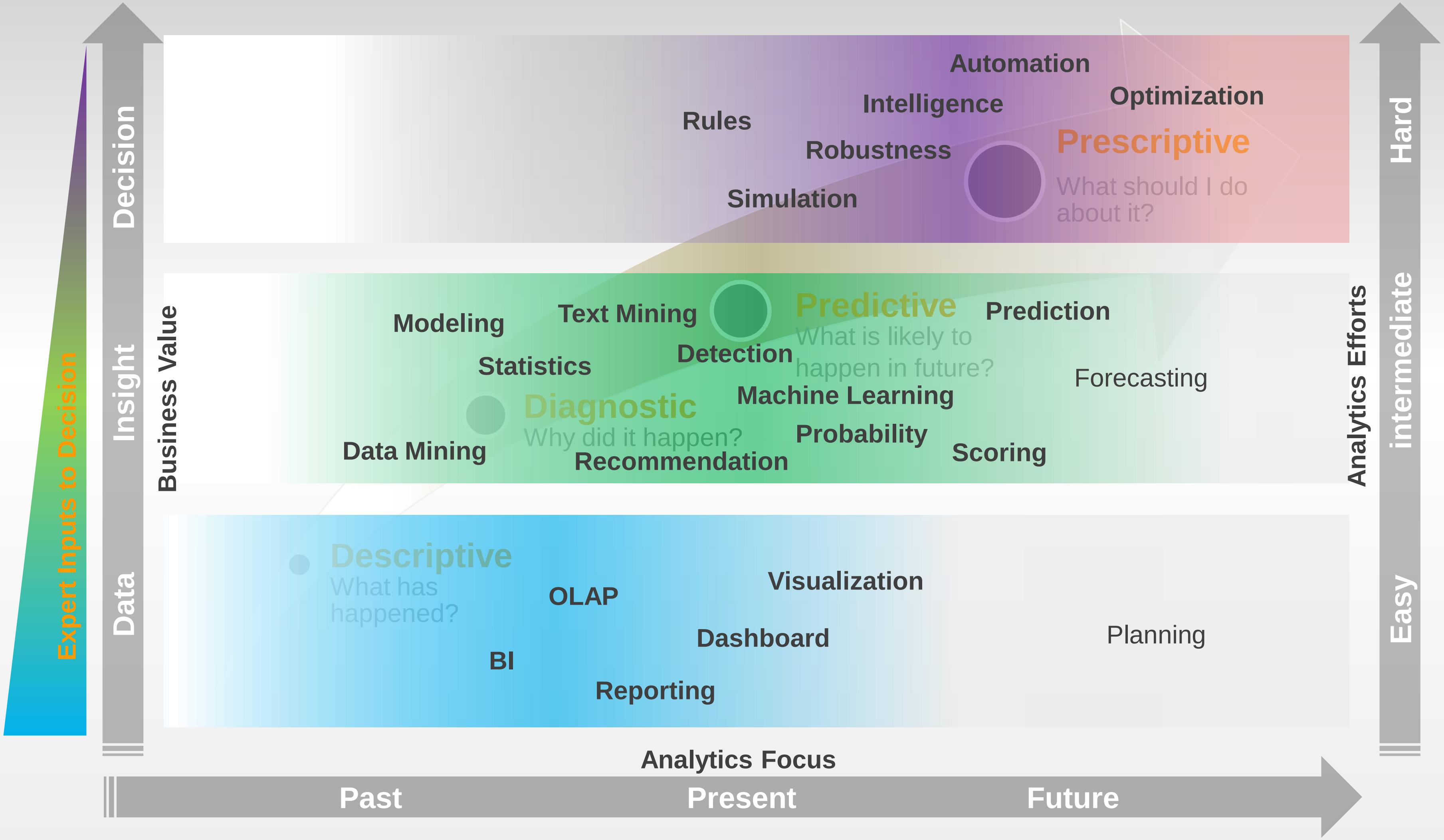


February, 2018
R4.1

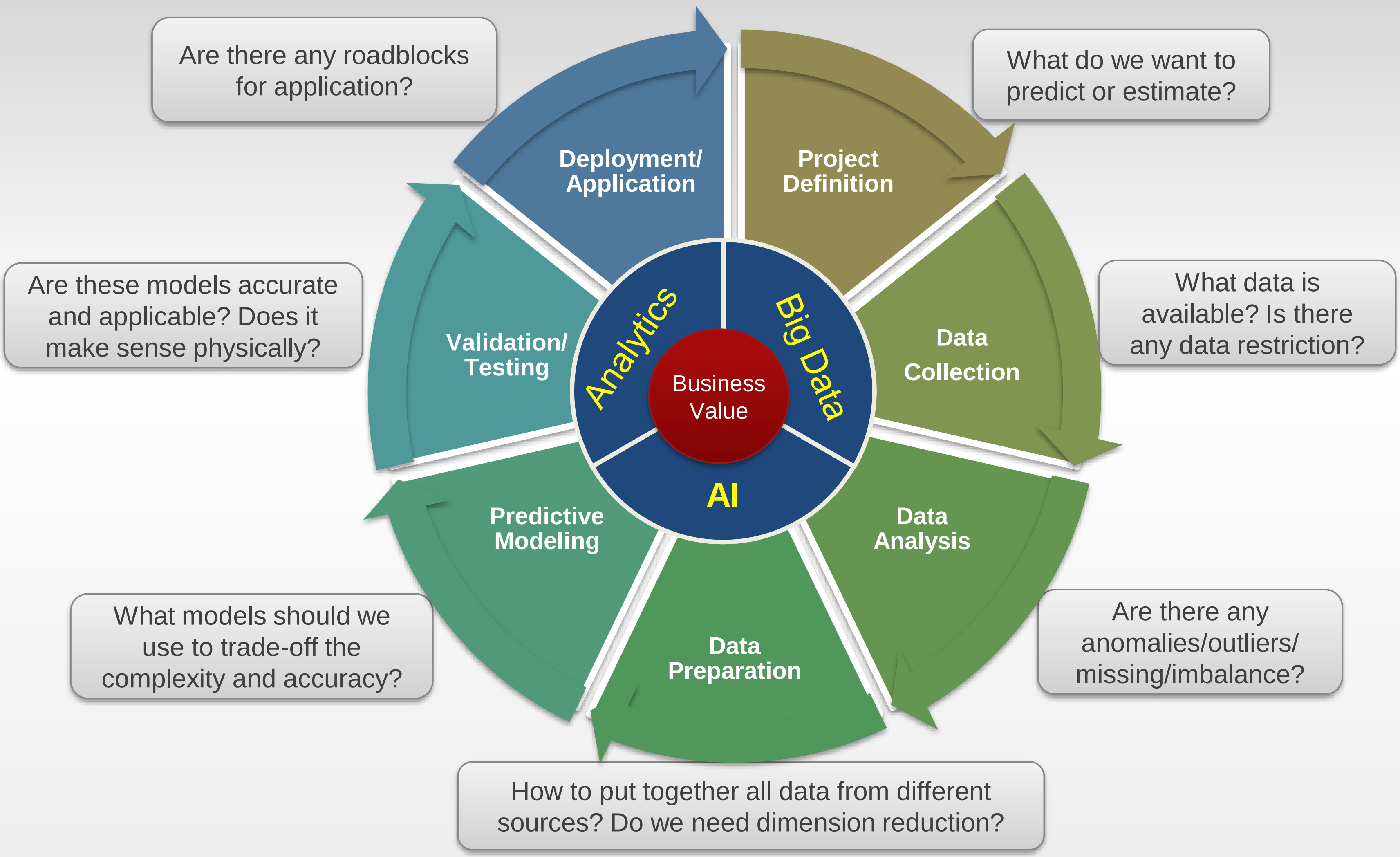
Version Control

Ver	Description	Date
1	Preliminary modeling • Customer historical pattern … Sales + Claims • Jan-Oct 2017 • Voting model (RF, DT, SVC) … 70% accuracy	12/10/2017
2	• Customer pattern (Sales, Claims) + **Product Attributes** (Size, Style, Color & Backing) • **Online application** w/ dashboard update on 1/10/2018 • Review with **Operation team** on 1/26/2018 • Model accuracy … 72%	1/10/2018
3	• Add ML/AI slides & online implementation flowchart • Compare models w/ or w/o PCA • **Segmentation**: Highly likely (>60%)..>80% accuracy, unsure, highly unlikely (<40%) • Review it in **Lunch & Learn**	2/2/2018
4	• Add product historical issues via rolls • New data set with cleaning in SQL script .. Jan 2017 – Feb 2018 • Explore more classification methods • Updated model accuracy … 83% • Segmentation: Highly likely (>75%)-->92%, unsure, highly unlikely (<25%)--~88% … cover 67% claims	2/22/2018

Analytics … Turn Data into Decision



Predictive Modeling … Iterative Procedure

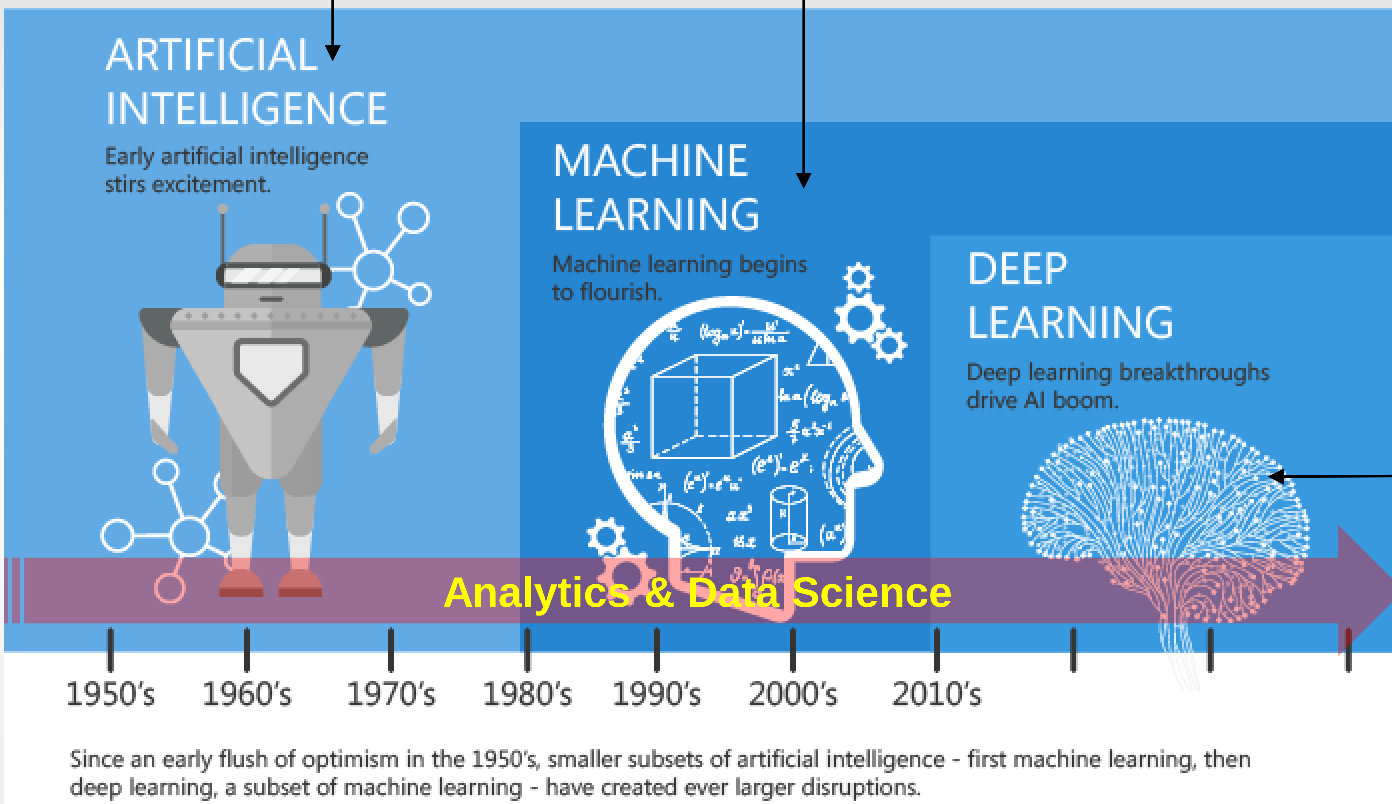


Analytics Techniques

…AI, Machine Learning & Deep Learning

Artificial Intelligence
Any technique which enables computers to mimic human behavior

Machine Learning
Subset of AI techniques which enables machines intelligent



http://blog.devitpl.com/learning-machine-learning/

Deep Learning
Subset of ML which makes the computation of complicated multi-layer neural networks feasible

- Big Data
- Complexity
- High Performance Computation
- Cloud technology
- High speed internet

Claim AI Scoring

Improve Claim Process Speed and Accuracy

Business Objective	Improve claim processing speed and accuracy, reduce cost of claims and improve customer satisfaction
Use Case Capability Description	Develop claim recommendation score to identify potential predictors of claims. Recommendation engine is based on customer claim history, claim to sales %, product and product claim history.

Use Case Sponsor	Claim Analysis	**Value Drivers**	• Costs – Claims spending

Current Challenges

- 100% of claims submitted require manual evaluation and high percentage of them require further inspection before decision is made
- Claim analysis reps spend a lot of time in navigating across multiple systems with incomplete information to investigate factors that determine validity of a claim
- Incorrect classification on a claim causing both unnecessary spend and efforts and impacting customer satisfaction
- Back log in claim processing due to need of manual evaluation in all claims

Analytics Insights & Solution Approach

Insights:

- Identify customer, product, manufacturing process attributes that are predictors of claims
- Provide timely and simple recommendation score on claims
- Identify shifts in production variables over time

Potential modeling approach:

- Propensity of false claims based on product and customer data, physical inspection is required.
- Propensity of valid claims with auto approval potential.

Impact on current processes:

- Reduce yearly claim spend with more insightful analysis on products, customers, and manufacturing history attributes while improving customer service through more timely processing of claims
- Reduce spend on claim processing with more accuracy, effectiveness and efficiency

Data Sources

- Customer Sales, Customer claims
- Product profile, Product sales and claim history
- Manufacturing profile

Claim AI Scoring - Overview

Data

- ✓ Modeling data: Jan 2017 - Feb, 2018
- ✓ Customer + Product + **Roll**
- ✓ 12 input factors
- ✓ 30,128 customer records for modeling
- ✓ Only claims related to quality
- ✓ Paid vs. Declined Claims = 71%: 29%

Method/Procedure

- ✓ Data cleaning: missing, imputation, etc.
- ✓ Data mining: dist. Box, outlier …
- ✓ Data normalization and preprocessing
- ✓ Machine learning: logistic, decision tree…
- ✓ Advanced analytics: Neural Network, Bayes, Deep Learning

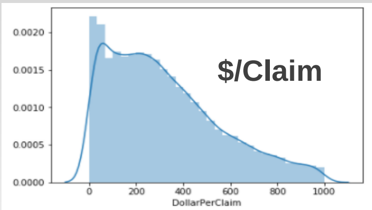
Results/Impacts

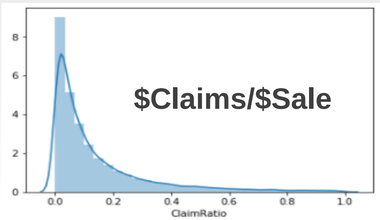
- ✓ Prediction Accuracy:
  - ✓ Overall: >83% (vs. 70% last model)
  - ✓ Highly likely: ~92% (vs. 80%)
  - ✓ Highly Unlikely: ~88% (vs. 75%)
- ✓ Potential reduction of claim processing effort by 60%+ (vs. 40%)
- ✓ Improve timeliness and accuracy in claim processing (hours/days to minutes)

Next

- ❑ Solution implementation/integration.
- ❑ Continuous model accuracy improvement
- ❑ Customer specific modeling

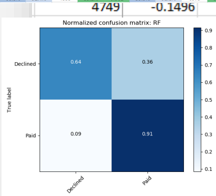
Modeling





Customer_GK	claimRatio	Claims	Pred	Score
512	-0.1695	1	1	0.738574
7526	-0.0577	1	1	0.738711
8419	-0.1388	1	1	0.738932
3334	-0.1160	1	1	0.738981
7344	-0.1584	1	1	0.739097
1697	-0.1236	0	1	0.73911
4516	-0.1213	1	1	0.739125
			1	0.73917
			1	0.739275
			1	0.739312
			1	0.73934
			1	0.73936
			1	0.739396
			1	0.739458
			1	0.739503
			1	0.739643
4749	-0.1496	1	1	0.739669
		1	1	0.739691
		0	1	0.739732
		1	1	0.739742
		1	1	0.739906
		1	1	0.739997
		1	1	0.740255
		1	1	0.740267
		1	1	0.74038

All inputs	Customer + Product + Rolls		1/2017-2/2018							
	Validation (6327)					Testing (9039)				
Model	TN (1816)	TP (4511)	Pred. Paid	Accu	CK_score	TN (2594)	TP (6445)	Pred. Paid	Accu	CK_score
L1_logistic	51.9%	72.7%	4153	66.7%	0.233	53.2%	72.4%	5882	66.9%	0.240
L2_logistic(OvR)	51.8%	72.5%	4146	66.5%	0.229	53.1%	72.3%	5875	66.8%	0.238
L2_logistic(Multinomial)	51.8%	72.7%	4154	66.7%	0.231	53.0%	72.4%	5888	66.9%	0.239
LinearSVC	41.9%	79.6%	4645	68.7%	0.219	43.7%	79.1%	6556	68.9%	0.230
RBFSVC	67.0%	79.8%	4200	76.1%	0.446	65.9%	79.2%	5988	75.4%	0.428
SGD	31.8%	68.2%	4316	57.8%	0.000	31.7%	68.7%	6197	58.1%	0.004
NaiveBayes	15.4%	95.5%	5843	72.5%	0.139	16.9%	95.3%	8299	72.8%	0.155
BGM	4.6%	93.0%	5929	67.6%	-0.031	4.3%	93.4%	8500	67.8%	-0.031
NearestNeighbors	55.8%	77.7%	4308	71.4%	0.324	55.4%	76.8%	6104	70.6%	0.310
DecisionTree	60.6%	87.0%	4640	79.4%	0.487	60.1%	87.0%	6642	79.3%	0.482
EDT	45.6%	77.2%	4471	68.1%	0.227	46.8%	77.7%	6386	68.8%	0.244
RandomForest	64.3%	91.2%	4761	83.4%	0.578	64.3%	91.3%	6811	83.5%	0.580
NueralNetwork	10.3%	99.5%	6116	73.9%	0.133	10.8%	99.4%	8720	73.9%	0.137
AdaBoost	47.0%	95.1%	5254	81.3%	0.480	47.9%	94.3%	7429	81.0%	0.476
GBRT	48.3%	95.0%	5223	81.6%	0.490	49.2%	94.4%	7402	81.4%	0.490
QDA	16.6%	93.2%	5720	71.3%	0.123	17.2%	93.1%	8148	71.3%	0.129
Bag	66.2%	88.9%	4622	82.4%	0.561	65.5%	89.4%	6656	82.6%	0.563
Voting 1 (DT, Bag, RF)	64.8%	91.3%	4758	83.7%	0.585	65.0%	91.2%	6788	83.7%	0.585
Voting 2 (SVC, Bag, RF)	60.3%	92.4%	4888	83.2%	0.562	59.9%	92.1%	6975	82.9%	0.554

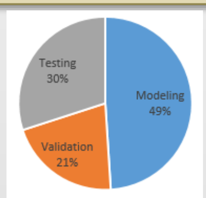


Input Factors

- #Claims
- #Sales
- $Claim
- $/Sale
- $Claim/$Sale
- Product Style
- Product Size
- Product Back
- Product Color
- F1ROLL
- F1GROL
- F1DLOT

Data (30128)

- Modeling: 14,762(49%)
- Validation:6,327(21%)
- Testing:9,019(30%)



Data Normalization
AI Modeling

17 Methods

L1 Logistic
L2 Logistic (OvR)
RBF SVC
Linear SVC
L2 Logistic (Multinomial)
Naïve Bayes
Nearest Neighbors
Decision Tree
Random Forest
Neural Network
AdaBoost
QDA
Extra Decision Tree
SGD
QDA
Bag
BGM

Output

- Paid (1) or Declined (0)
- Confidence (Score)

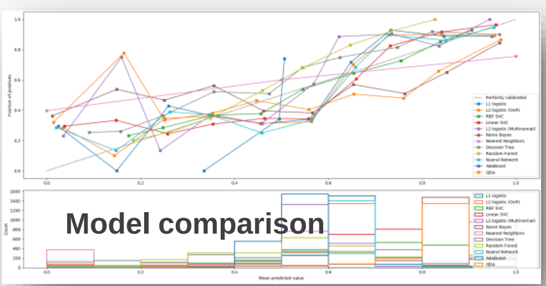
- **Target: binary classification**
  Paid (1) vs. Declined (0)
- **Data Components**
  Paid: 71.3%
  Declined: 28.7%

```
                    Claims  numClaims  dollarClaims  numSales  dollarSales  \
PRIMARY_CUSTOMER_GK
55116                    1          1         46.00      10.0    241296.53
55121                    1          2       4675.67      10.0    112419.43
55213                    1         12      13445.83       8.0     42637.88
55310                    0          1       2278.70      11.0    633067.71
55602                    1          1        815.52      10.0   1088677.47

                                     CUSTOMER_NAME  dollarPerClaim  \
PRIMARY_CUSTOMER_GK
55116                             CB FLOORING LLC       46.000000
55121                             CB FLOORING LLC     2337.835000
55213              COMMERCIAL CPTS OF AMERICA INC     1120.485833
55310                     FAMILY CARPET OUTLET INC     2278.700000
55602                              NORDSTROMS INC      815.520000

                    dollarPerSale  claimRatio
PRIMARY_CUSTOMER_GK
55116                   24129.653    0.000191
55121                   11241.943    0.041591
55213                    5329.735    0.315349
55310                   57551.610    0.003599
55602                  108867.747    0.000749
```

Sample data



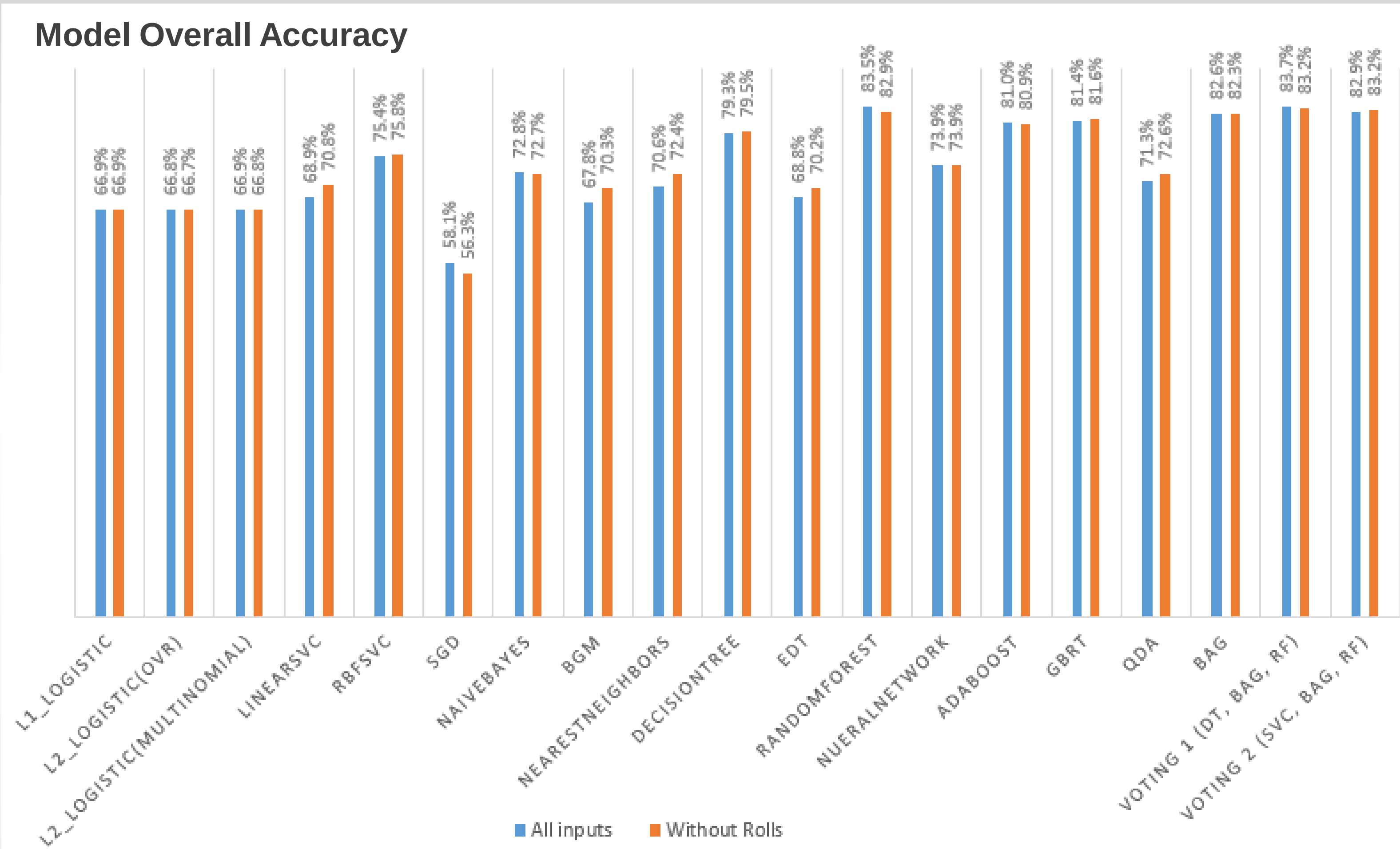
Model comparison

Results – Customer + Product + Rolls

All inputs	Customer + Product + Rolls			1/2017-2/2018						
	Validation (6327)					Testing (9039)				
Model	TN (1816)	TP (4511)	Pred. Paid	Accu	CK_score	TN (2594)	TP (6445)	Pred. Paid	Accu	CK_score
L1_logistic	51.9%	72.7%	4153	66.7%	0.233	53.2%	72.4%	5882	66.9%	0.240
L2_logistic(OvR)	51.8%	72.5%	4146	66.5%	0.229	53.1%	72.3%	5875	66.8%	0.238
L2_logistic(Multinomial)	51.8%	72.7%	4154	66.7%	0.231	53.0%	72.4%	5888	66.9%	0.239
LinearSVC	41.9%	79.6%	4645	68.7%	0.219	43.7%	79.1%	6556	68.9%	0.230
RBFSVC	67.0%	79.8%	4200	76.1%	0.446	65.9%	79.2%	5988	75.4%	0.428
SGD	31.8%	68.2%	4316	57.8%	0.000	31.7%	68.7%	6197	58.1%	0.004
NaiveBayes	15.4%	95.5%	5843	72.5%	0.139	16.9%	95.3%	8299	72.8%	0.155
BGM	4.6%	93.0%	5929	67.6%	-0.031	4.3%	93.4%	8500	67.8%	-0.031
NearestNeighbors	55.8%	77.7%	4308	71.4%	0.324	55.4%	76.8%	6104	70.6%	0.310
DecisionTree	60.6%	87.0%	4640	79.4%	0.487	60.1%	87.0%	6642	79.3%	0.482
EDT	45.6%	77.2%	4471	68.1%	0.227	46.8%	77.7%	6386	68.8%	0.244
RandomForest	64.3%	91.2%	4761	83.4%	0.578	64.3%	91.3%	6811	83.5%	0.580
NueralNetwork	10.3%	99.5%	6116	73.9%	0.133	10.8%	99.4%	8720	73.9%	0.137
AdaBoost	47.0%	95.1%	5254	81.3%	0.480	47.9%	94.3%	7429	81.0%	0.476
GBRT	48.3%	95.0%	5223	81.6%	0.490	49.2%	94.4%	7402	81.4%	0.490
QDA	16.6%	93.2%	5720	71.3%	0.123	17.2%	93.1%	8148	71.3%	0.129
Bag	66.2%	88.9%	4622	82.4%	0.561	65.5%	89.4%	6656	82.6%	0.563
Voting 1 (DT, Bag, RF)	64.8%	91.3%	4758	83.7%	0.585	65.0%	91.2%	6788	83.7%	0.585
Voting 2 (SVC, Bag, RF)	60.3%	92.4%	4888	83.2%	0.562	59.9%	92.1%	6975	82.9%	0.554

- Recommend models: **RBF-SVC**, **Decision Tree, Random Forest & Bag**, balancing positive & negative true with balanced classification
- SVC requires tuning up the parameters C and gamma
- **Voting** method may not be needed as individual models perform well already

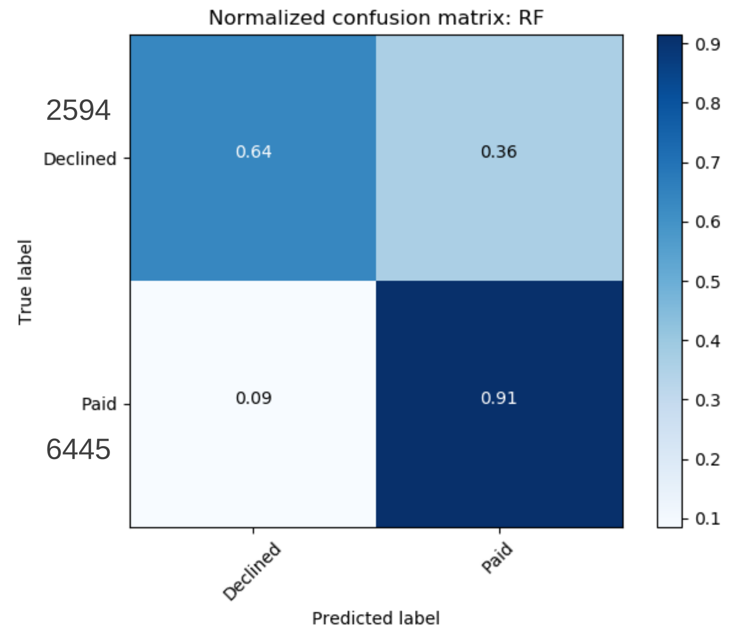
Model Accuracy Comparison (w/o Rolls)



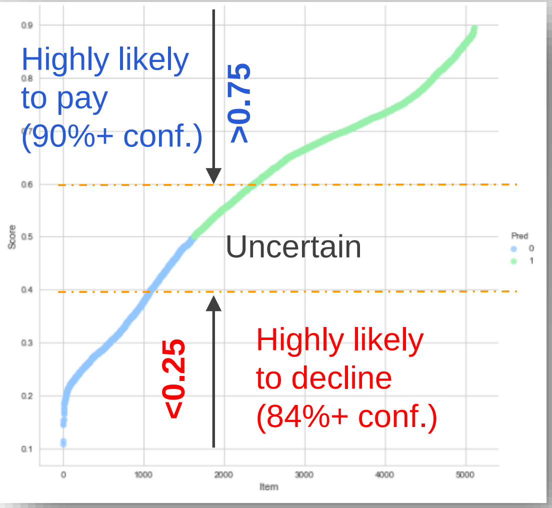
The impact of historical issues of a roll on the overall accuracy is Not significant

Results

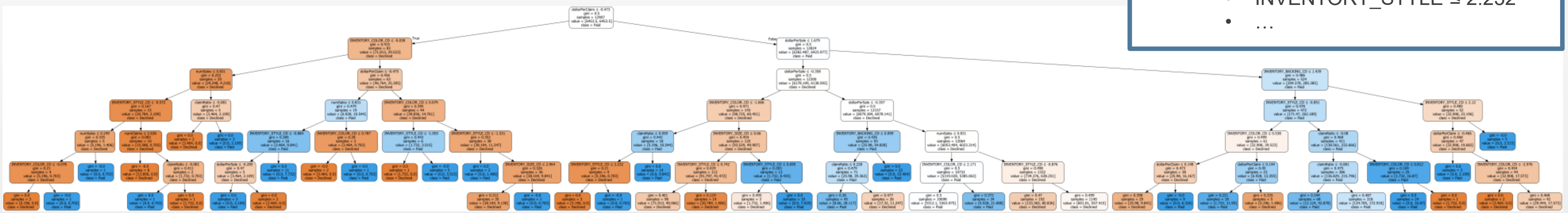
Confusion Matrix



AI Scoring (0, 1)



Decision Tree

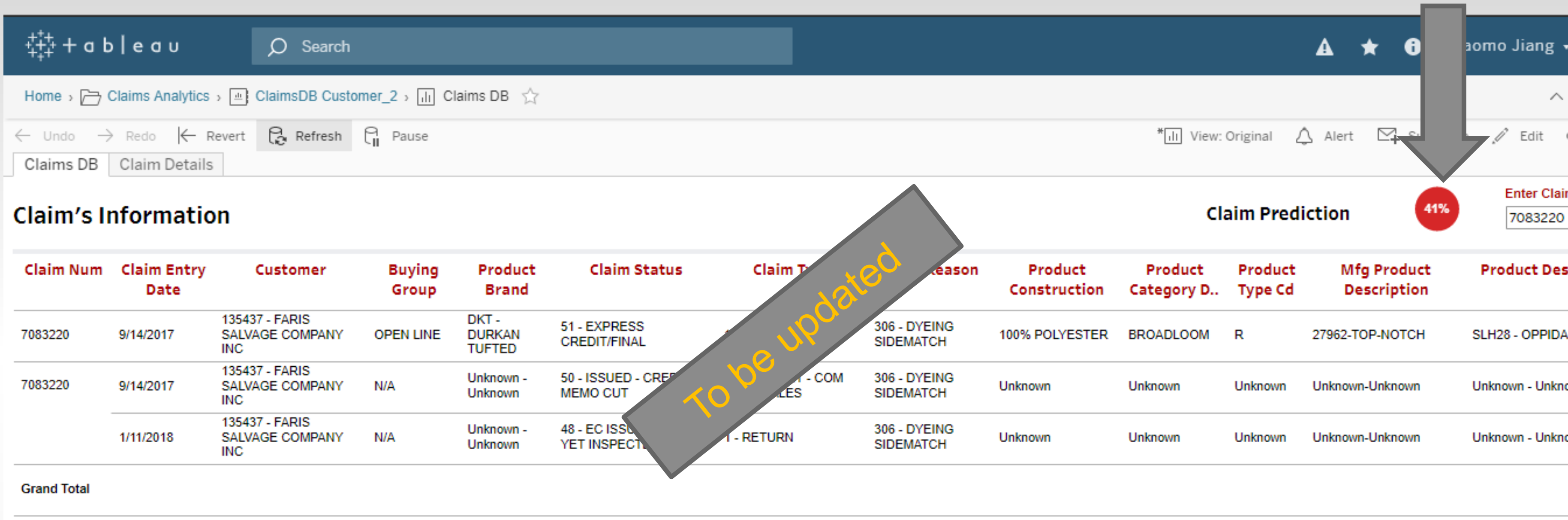


Scoring Decision Logic

- Level 1
  - dollarPerClaim ≤ -0.473
- Level 2
  - INVENTORY_COLOR ≤ -0.038
  - dollarPerSale ≤ 1.679
- Level 3
  - numSales ≤ 0.831
  - dollarPerClaim ≤ -0.475
  - dollarPerSale ≤ -0.358
  - INVENTORY_BACKING ≤ 1.439
- Level 4
  - INVENTORY_STYLE ≤ -0.572
  - claimRatio ≤ -0.081
  - numSales ≤ 0.831
  - INVENTORY_COLOR ≤ 0.079
  - INVENTORY_COLOR ≤ -1.666
  - dollarPerSale ≤ -0.357
  - INVENTORY_STYLE ≤ -0.851
  - INVENTORY_STYLE ≤ 2.12
- Level 5
  - numSales ≤ 0.249
  - numClaims ≤ 2.636
  - INVENTORY_STYLE ≤ -0.884
  - INVENTORY_COLOR ≤ 0.787
  - INVENTORY_STYLE ≤ -1.055
  - INVENTORY_STYLE ≤ -1.331
  - …
- Level 6
  - dollarPerSale ≤ -0.285
  - INVENTORY_SIZE ≤ 2.964
  - INVENTORY_STYLE ≤ 2.232
  - …

AI Score in Dashboard

Declined (< 50% prob. to pay)



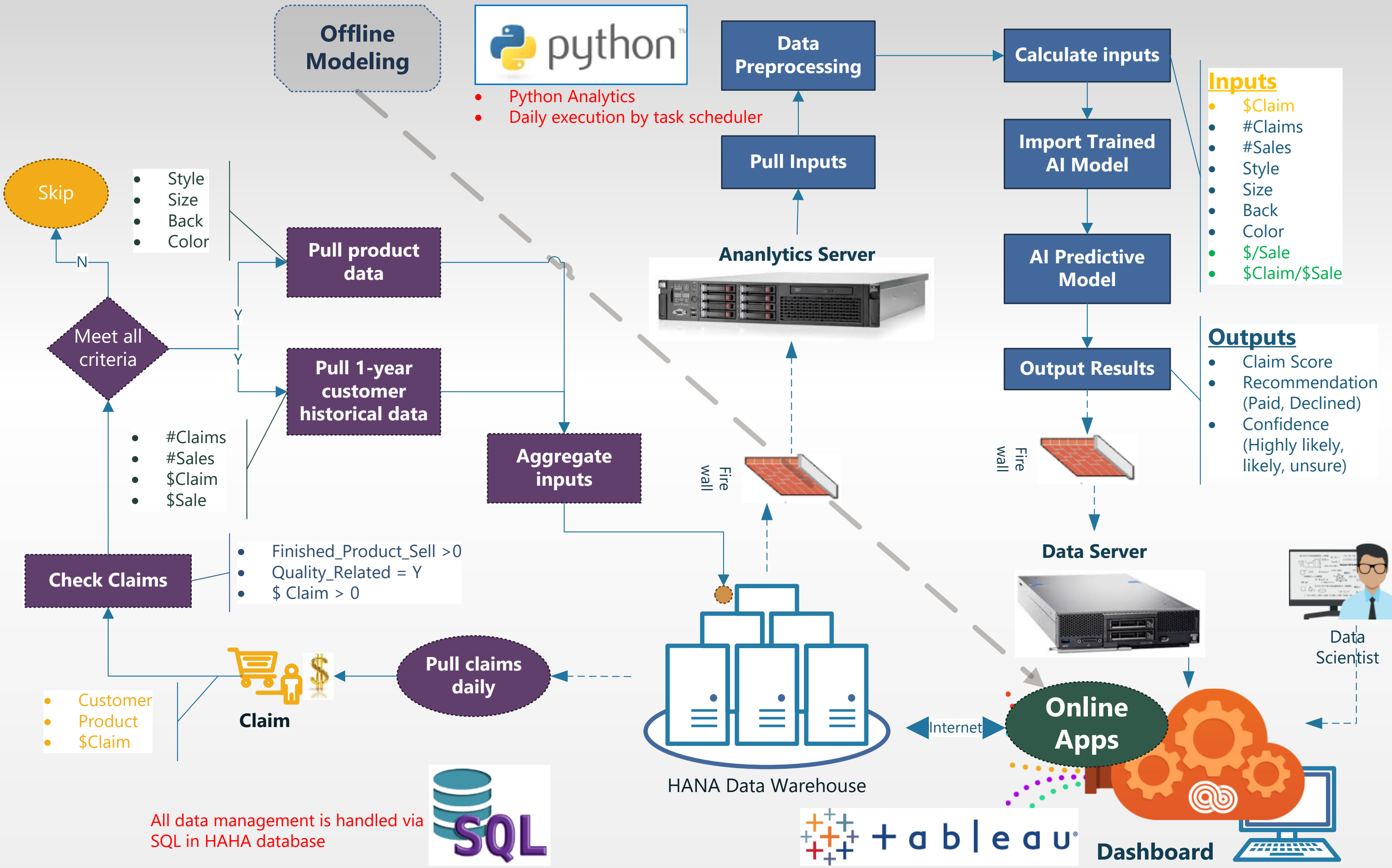
Next…

- ❑ Solution implementation/integration…dashboard update
- ❑ Continuous model accuracy improvement
- ❑ Customer specific modeling

Thanks

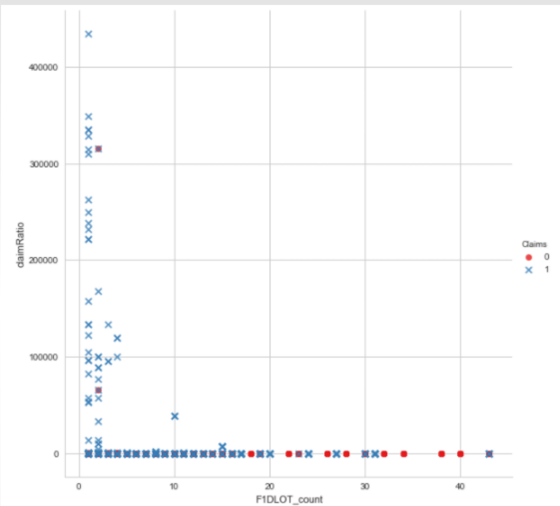
MOHAWK

Online Application



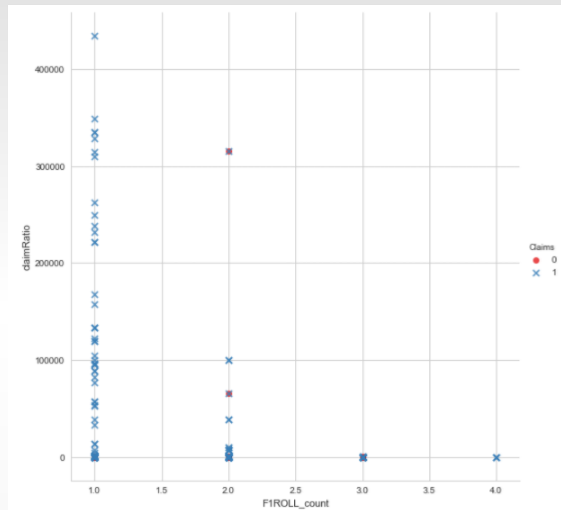
Influencing Factors for Claims (30,000+)

F1DLOT count



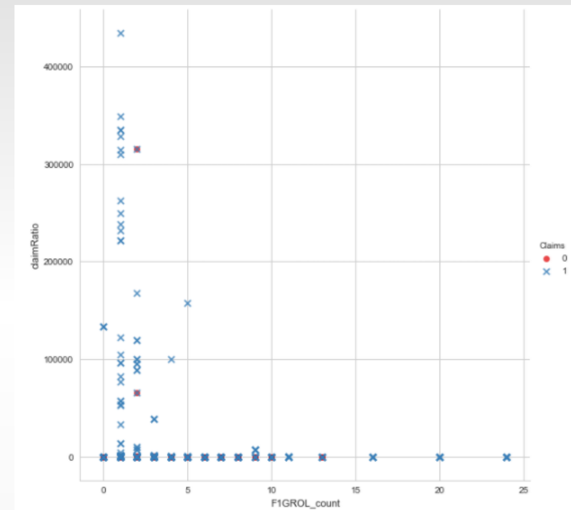
4849 F1DLOTs have >1 claims
(16% data)

F1ROLL count



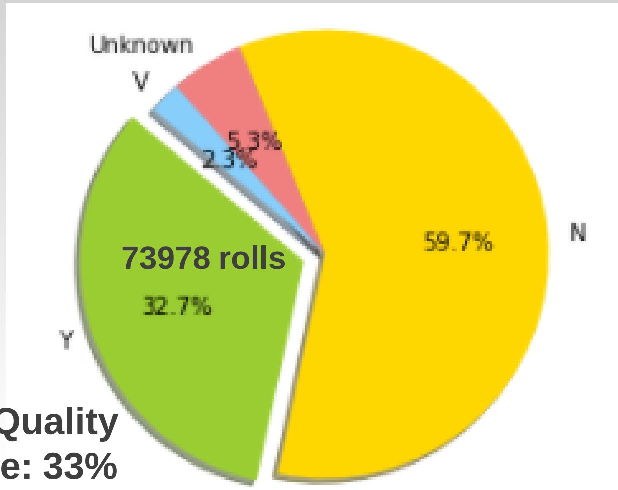
1372 F1ROLLs have >1 claims
(4.6% data)

F1GROL count

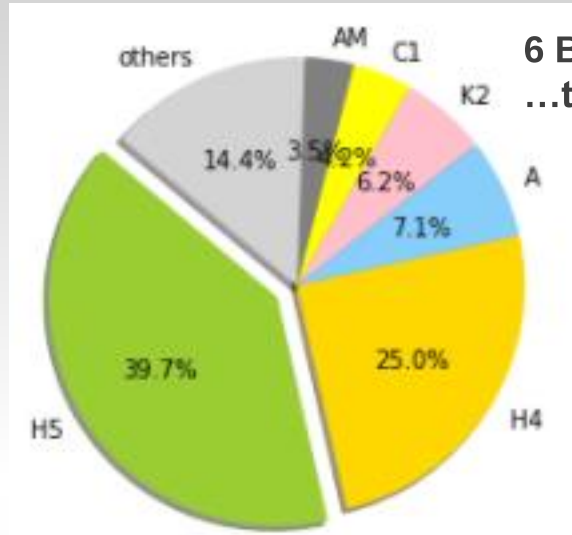


3951 F1GROLs have >1 claims
(13% data)

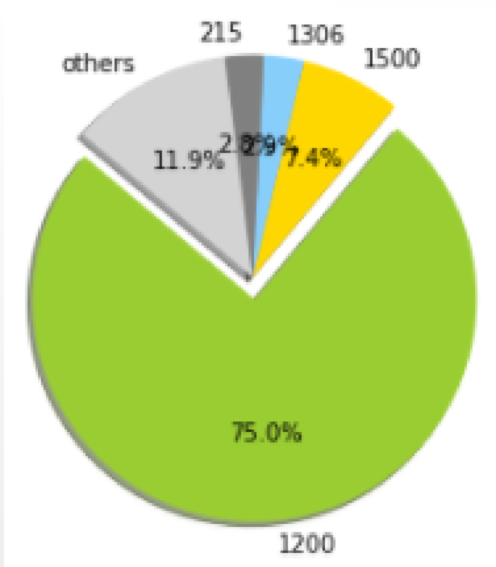
Influencing Factors for Claims



Y = Quality issue: 33%

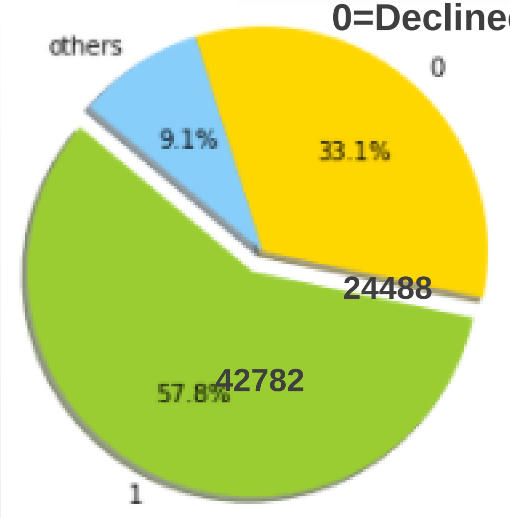


6 Backing types: 85%+
…total 89 backs



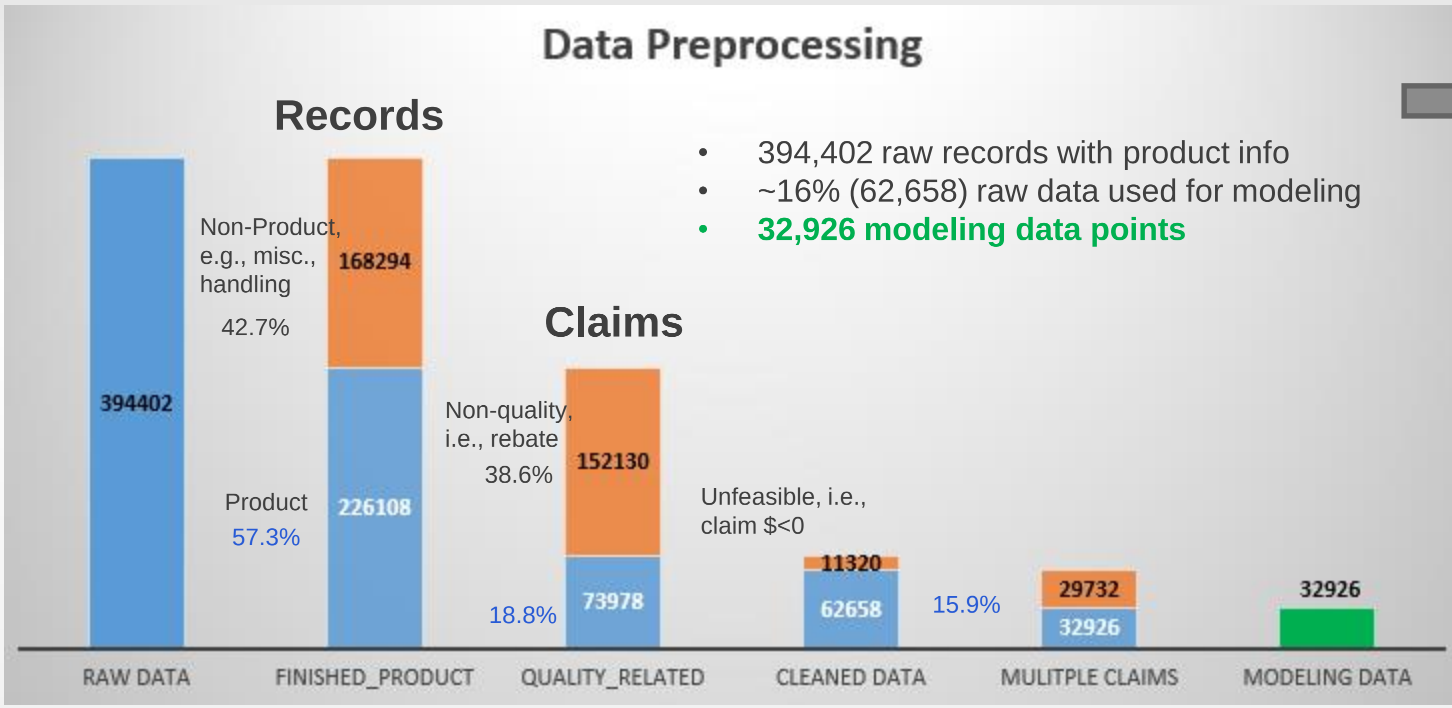
4 Sizes: 88%+
Total 166

1 = Paid,
0=Declined

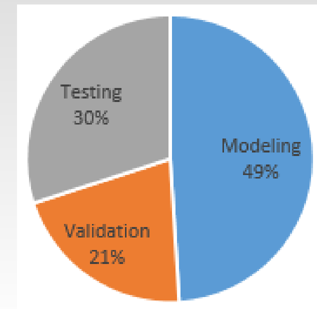


MOHAWK

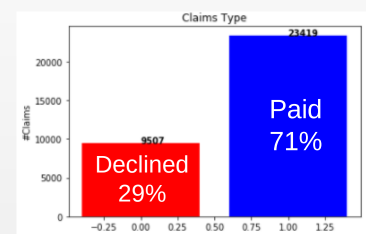
Modeling Data (old)



- 394,402 raw records with product info
- ~16% (62,658) raw data used for modeling
- **32,926 modeling data points**



Modeling	16133
Validation	6915
Testing	9878



Results (R2) – PCA vs. Raw

	Validation (6915)				Testing (9878)				Comments
Actual	Accuracy	Pred. Paid (1)	4918(1)	1997(0)	Accuracy	Pred. Paid (1)	7026(1)	2852(0)	
L1 Logistic	71.8%/72.1%	6598/6651	97%/98%	9%/8%	72.1%/72.3%	9482/9537	98%/98%	9%/8%	Prone to Paid, ↑false , reduce $
L2 Logistic (OvR)	71.8%72.2%	6597/6603	97%/98%	9%/10%	72.1%/72.4%	9482/9469	98%/98%	9%/9%	Prone to Paid, ↑false , reduce $
RBF SVC	66.4%/66.3%	4581/4671	73%/74%	50%/48%	65.7%/66.0%	6518/6653	72%/73%	49%/48%	Slightly balanced, but ↑false
Linear SVC	65.8%/67.4%	4920/5256	76/80%	41%/35%	66.0%/67.9%	6931/7422	75%/80%	43%/38%	Prone to Paid, ↑false , reduce $
L2 Logistics (Multinormial)	71.8%/72.2%	6597/6581	97%/95%	9%/10%	72.1%/72.4%	9482/9437	98%/98%	9%/10%	Prone to Paid, ↑false , reduce $
Naïve Bayes	69.1%/70.0%	6170/6252	91%/92%	15%/15%	69.3%/70.8%	8781/8932	91%/93%	16%/16%	Prone to Paid, ↑false , reduce $
Nearest Neighbor	64.8%/57.3%	4679/3718	73%/58%	45%/56%	64.3%/57.3%	6644/5287	72%/58%	45%/57%	Prone to declined, ↑miss, ↓customer
Decision Tree	59.5%/66.1%	3650/4275	59%/70%	62%/57%	60.1%/66.4%	5215/6067	59%/70%	63%/59%	Balanced prediction results
Random Forest	66.1%/65.0%	4949/4781	77%/74%	41%/43%	66.2%/65.1%	7009/6721	76%/73%	42%/45%	Balanced, but slightly prone to Paid
Neural Network	71.6%/71.8%	6784/6735	99%/99%	4%/6%	71.6%/71.9%	9723/9656	99%/99%	3%/5%	Prone to Paid, ↑false , reduce $
AdaBoost	72.1%/72.6%	6471/6358	96%/95%	13%/17%	72.0%/72.9%	9297/9038	96%/95%	12%/18%	Prone to Paid, ↑false , reduce $
QDA	70.0%/70.7%	6301/6366	93%/94%	13%/13%	70.5%/71.2%	9009/9074	93%/94%	14%/14%	Prone to Paid, ↑false , reduce $
Voting(RF, BLAG, SVC)	70.0%69.3%	5529/5155	85%/81%	33%/41%	70.1%/70.5%	7839/7314	85%/(81%	34%/44%	Prone to Paid

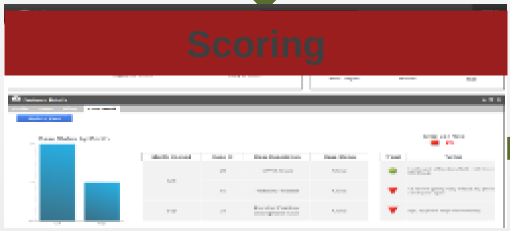
- PCA/Raw shown
- PCA with 95% info – 5 PCAs considered, vs. 9 raw factors
- **No significant improvement observed from PCA process**

Claim Scoring (Proposal)

- Customer
- Product
- Transactions
  Order – Ship - Bill
- Claims
- External
  Market, Consumer, Competitor
- Social
  Industry Reviews, Trends

Customer Analytics

Behavioral Segmentation	Propensity To Buy
Customer Claims	Customer sales
Claims / Sales Pattern	Claims / Product Pattern



Recommendation

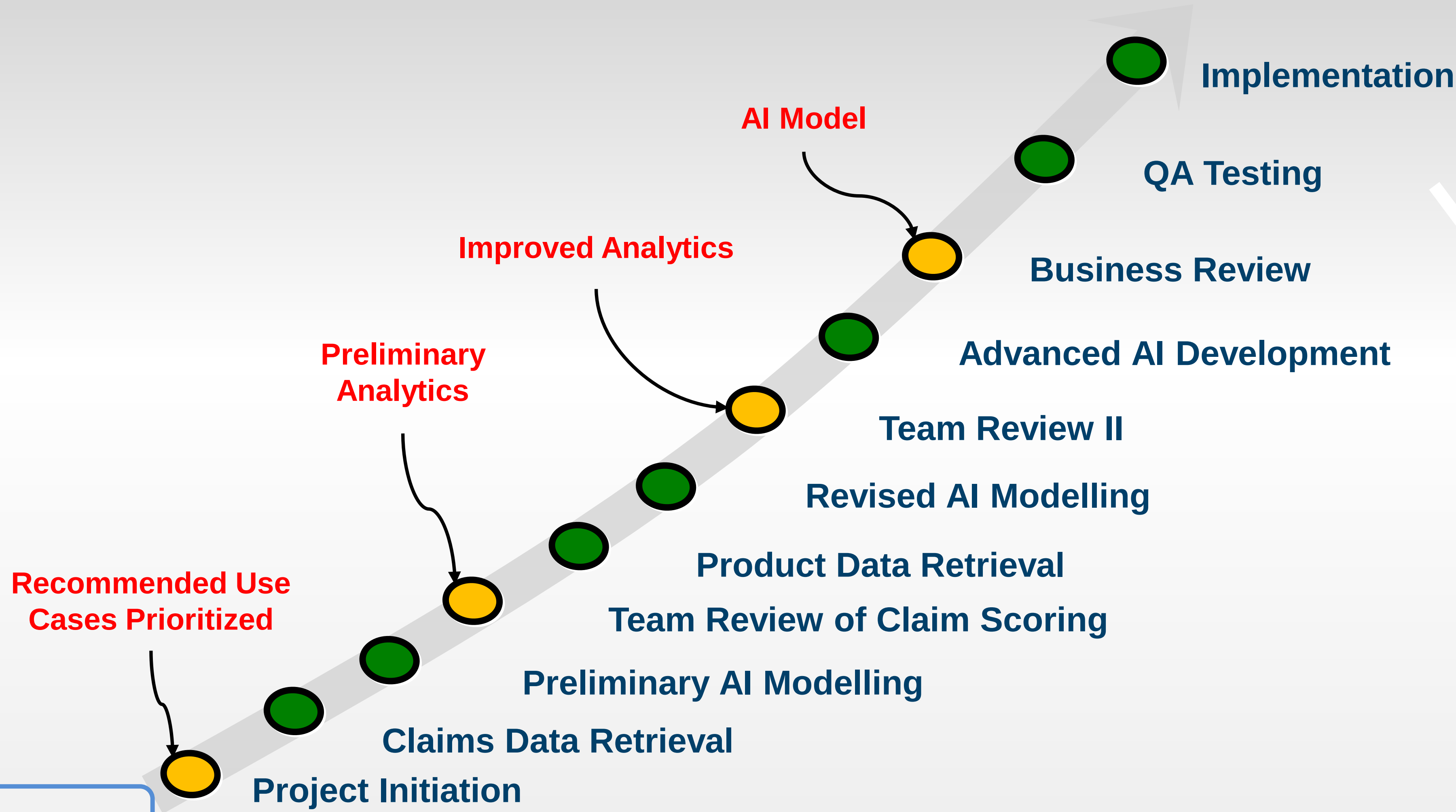
KPI, Dashboards

Worksheets

Operational Systems

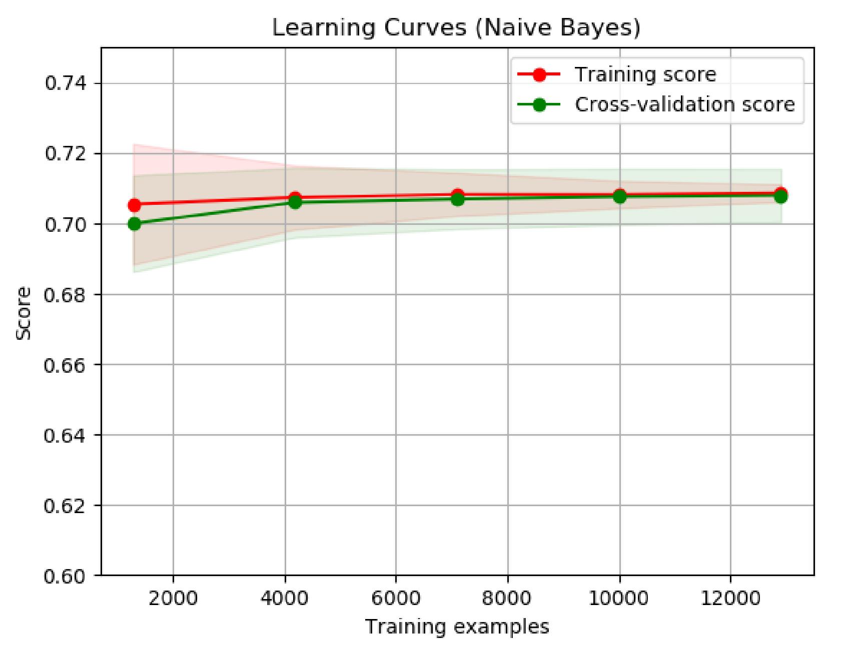
- Campaigns
- Sales Mgmt
- Retail Portal
- Customer Service

Claim AI Scoring Analytics Roadmap

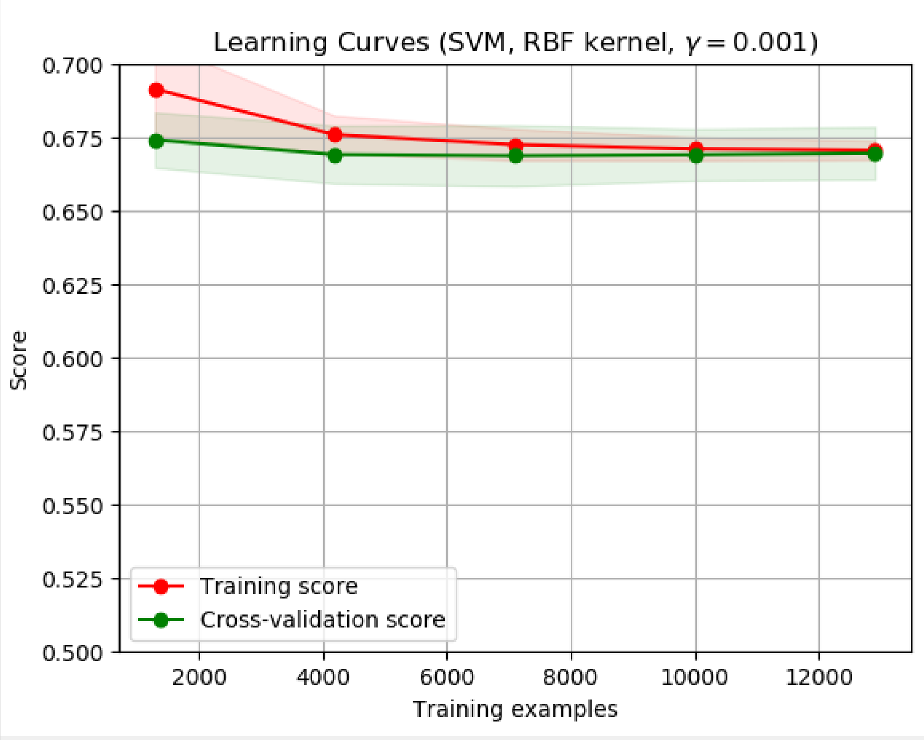


Roadmap

Learning Curves

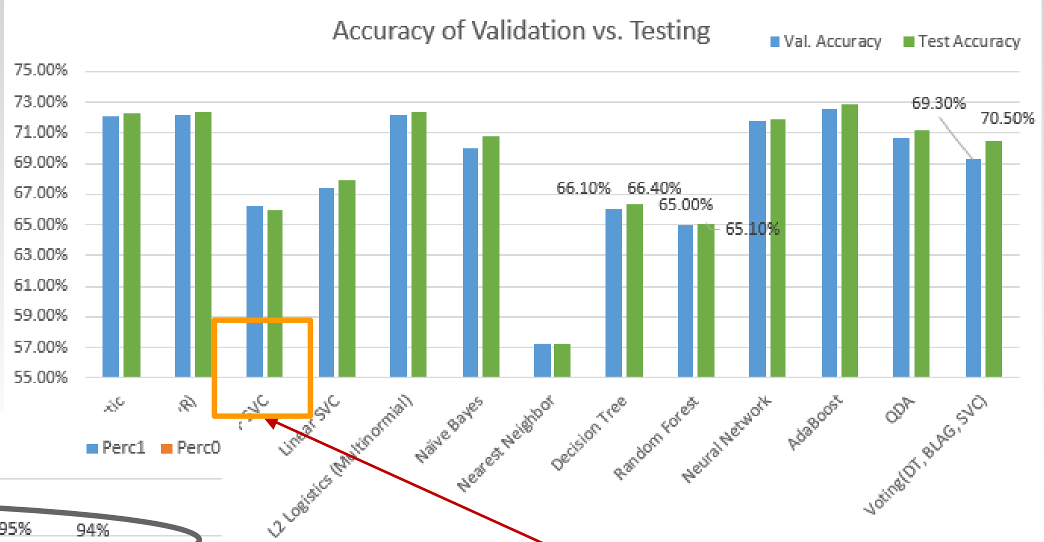


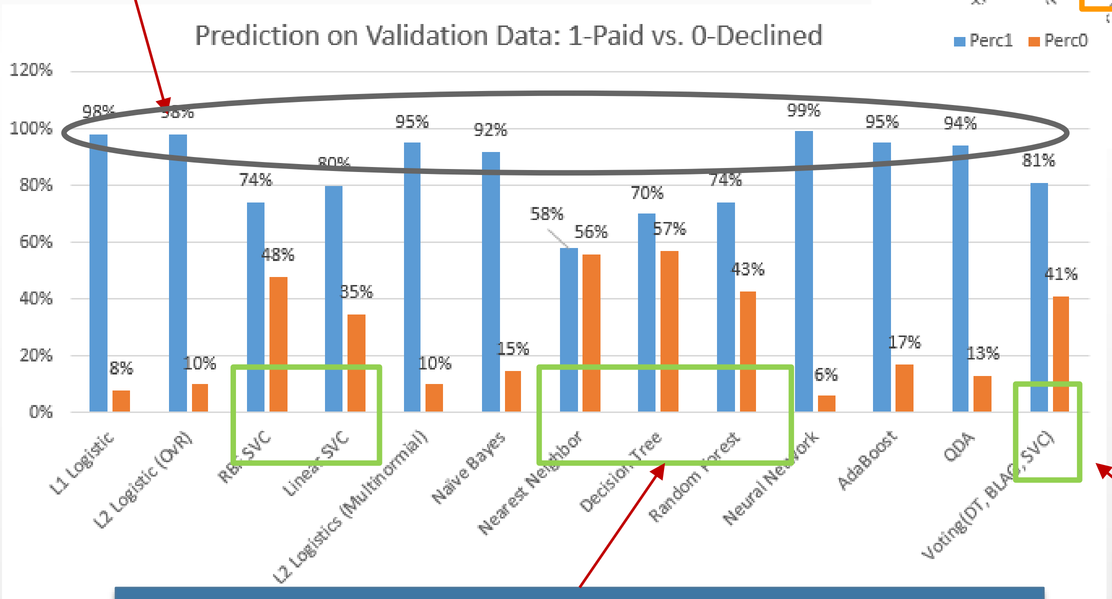
Fewer samples needed for Naïve Bayes method



Method Comparison (R2)

- In favor of predicting the "Paid" claims, company will loss benefits
- No significant improvement is obtained by adding product features





- Except for SVC, all other methods seem to have the testing accuracy larger than validation ones, which indicates that the models may be overfitting for 'Paid"…need to be further improved for business purpose

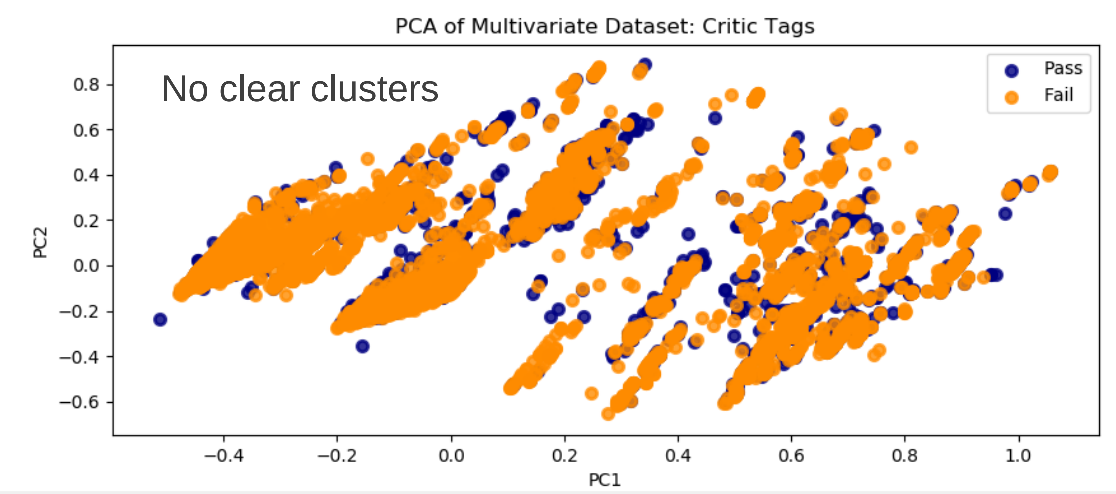
- SVC, NN, DT & RF: Balance the binary prediction.

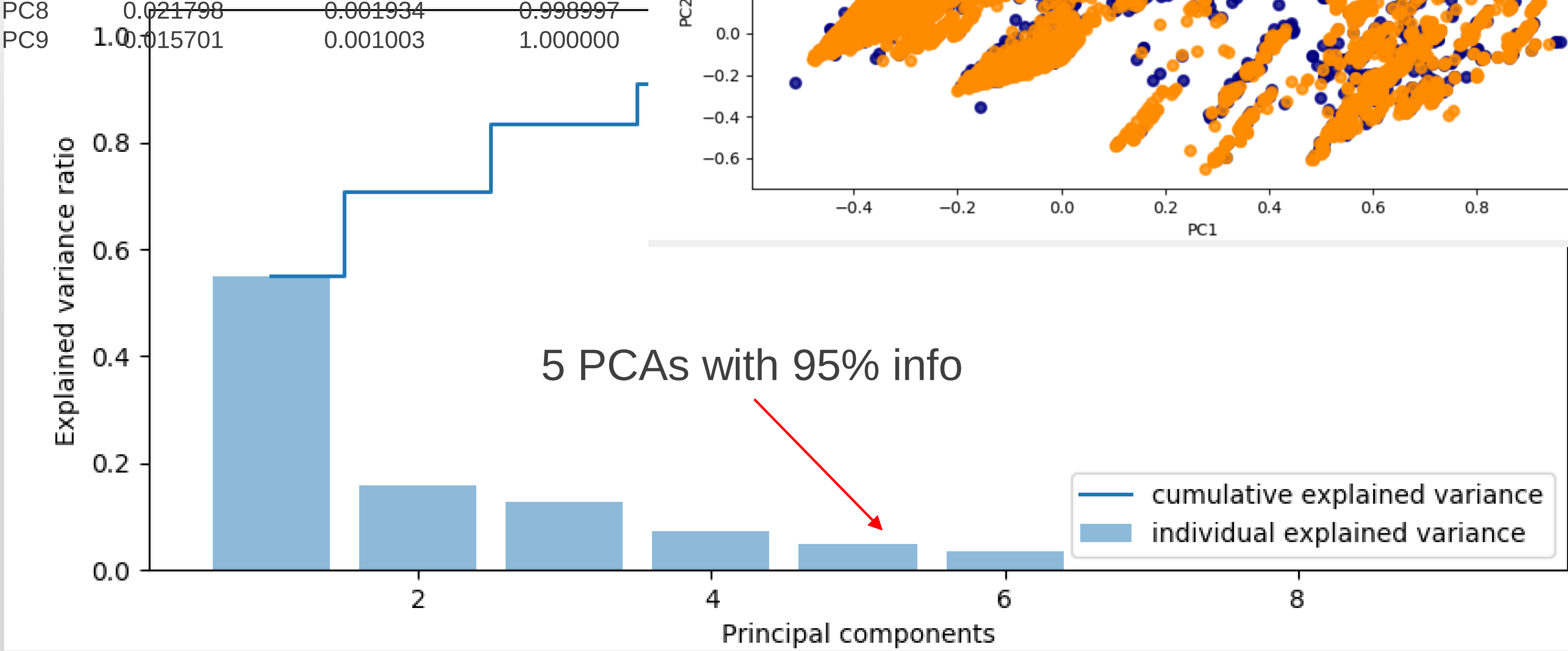
- Voting method: Balance the accuracy and bias of prediction.

PCA

PC/Standard deviation/Proportion of Variance/Cumulative Proportion

PC1	0.367471	0.549483	0.549483
PC2	0.196869	0.157711	0.707194
PC3	0.176534	0.126814	0.834008
PC4	0.135046	0.074211	0.908219
PC5	0.110978	0.050117	0.958336
PC6	0.095005	0.036728	0.995065
PC7	0.022163	0.001999	0.997063
PC8	0.021798	0.001934	0.998997
PC9	0.015701	0.001003	1.000000



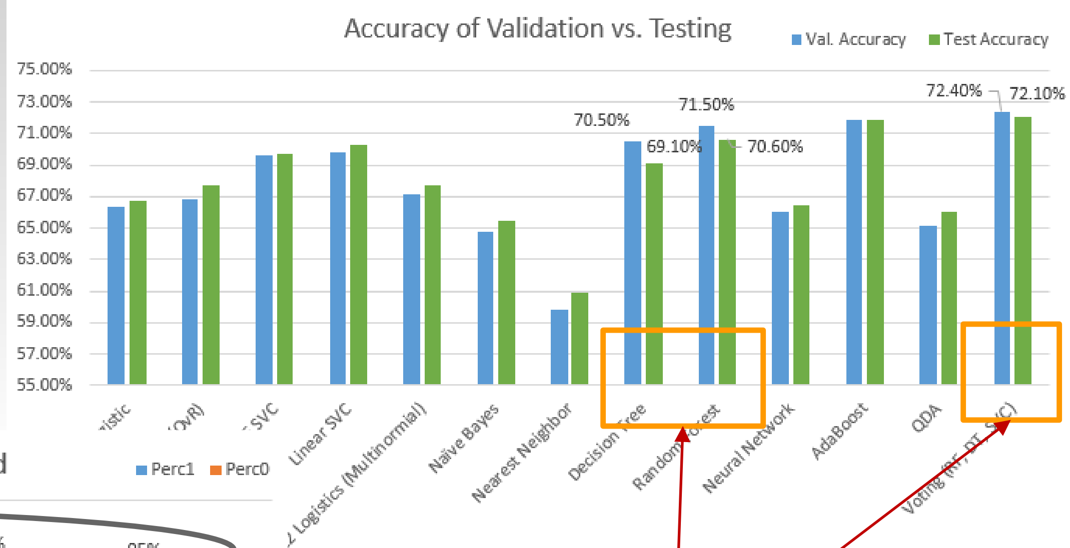


Results (R1) – Customer data only

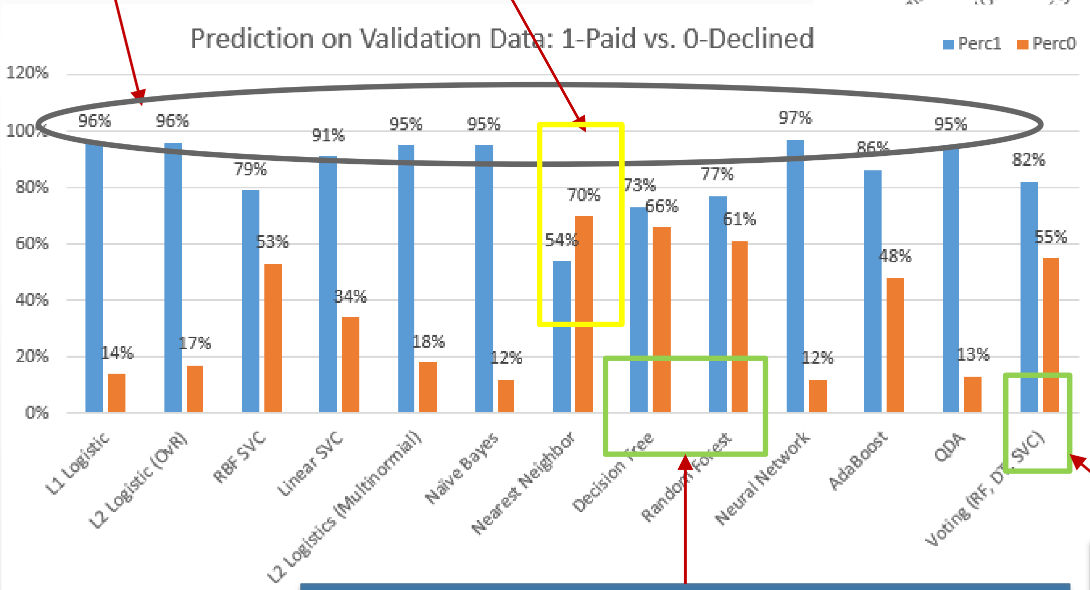
	Validation (2104)				Testing (3006)				Comments
Actual	Accuracy	Pred. Paid (1)	1335(1)	769(0)	Accuracy	Pred. Paid (1)	1908(1)	1098(0)	
L1 Logistic	66.4%	1943	1282(96%)	108(14%)	66.7%	2771	1832(96%)	165(15%)	Prone to Paid, ↑false , reduce $
L2 Logistic (OvR)	66.8%	1918	1282(96%)	131(17%)	67.7%	2740	1832(96%)	198(18%)	Prone to Paid, ↑false , reduce $
RBF SVC	69.6%	1418	1055(79%)	408(53%)	69.7%	1992	1488(78%)	604(55%)	Slightly balanced, but ↑false
Linear SVC	69.8%	1721	1215(91%)	261(34%)	70.3%	2446	1736(91%)	384(35%)	Prone to Paid, ↑false , reduce $
L2 Logistics (Multinormial)	67.1%	1902	1268(95%)	138(18%)	67.7%	2711	1832(96%)	209(19%)	Prone to Paid, ↑false , reduce $
Naïve Bayes	64.8%	1942	1268(95%)	92(12%)	65.5%	2772	1813(95%)	143(13%)	Prone to Paid, ↑false , reduce $
Nearest Neighbor	59.8%	958	721(54%)	538(70%)	60.9%	1385	1068(56%)	769(70%)	Prone to declined, ↑miss, ↓customer
Decision Tree	70.5%	1236	975(73%)	508(66%)	69.1%	1715	1355(71%)	725(66%)	Balanced prediction results
Random Forest	71.5%	1332	1028(77%)	469(61%)	70.6%	1896	1469(77%)	659(60%)	Balanced, but slightly prone to Paid
Neural Network	66.1%	1981	1295(97%)	92(12%)	66.5%	2834	1870(98%)	132(12%)	Prone to Paid, ↑false , reduce $
AdaBoost	71.9%	1546	1148(86%)	369(48%)	71.9%	2178	1622(85%)	538(49%)	Prone to Paid, ↑false , reduce $
QDA	65.2%	1937	1268(95%	100(13%)	66.1%	2769	1832(96%)	154(14%)	Prone to Paid, ↑false , reduce $
Voting(RF, DT, SVC)	72.4%	1440	1095(82%)	423(55%)	72.1%	2045	1565(82%)	615(56%)	Prone to Paid

- Recommend two models: **Decision Tree** and **Random Forest**
- Both balance the paid and declined prediction, particularly DT method
- **Voting** method is strongly recommended if possible, as it is robust after integrating different individual models

Method Comparison (R1) – Customer Data Only



- Prone to predict the “Paid” claims, company will loss benefits
  - NN method: Prone to predict the “Declined” claims, customer will not be happy

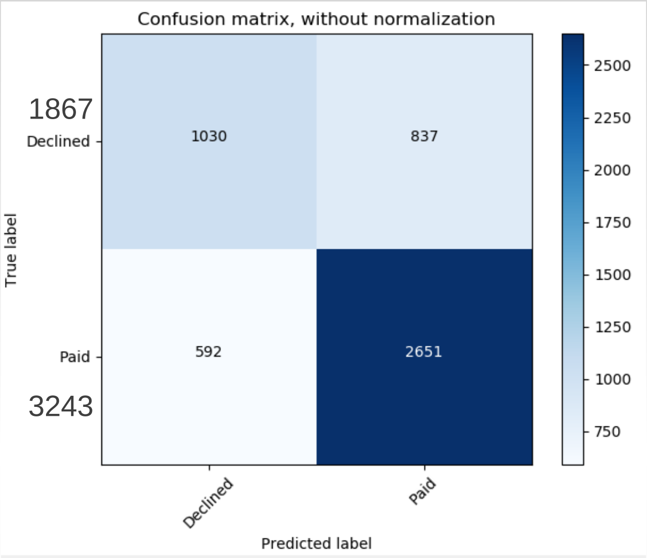


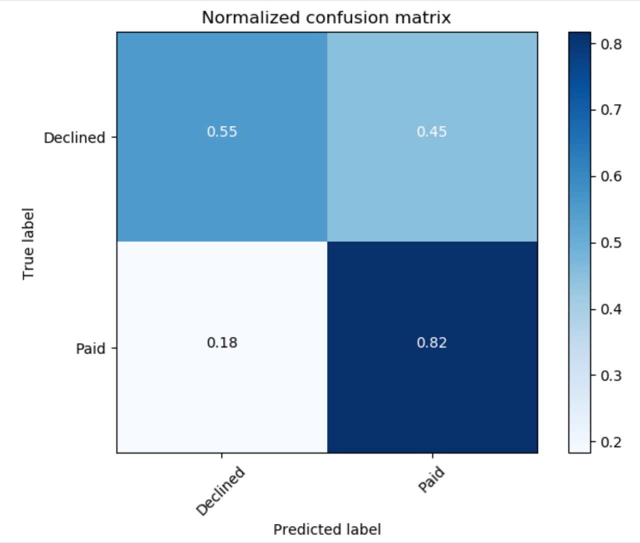
- DT, RF & Voting methods have the testing accuracy lower than validation ones, which makes sense
- Other methods may be overfitting for ‘Paid”…need to be improved for business purpose

- DT & RF: Balance the binary prediction.

- Voting method: Balance the accuracy and bias of prediction.

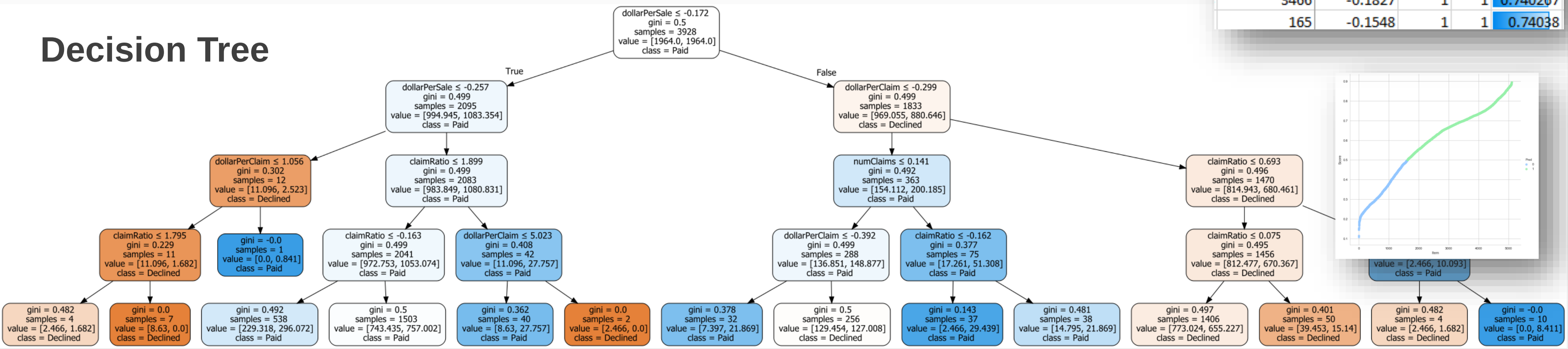
Visualization (R1) – Customer Data



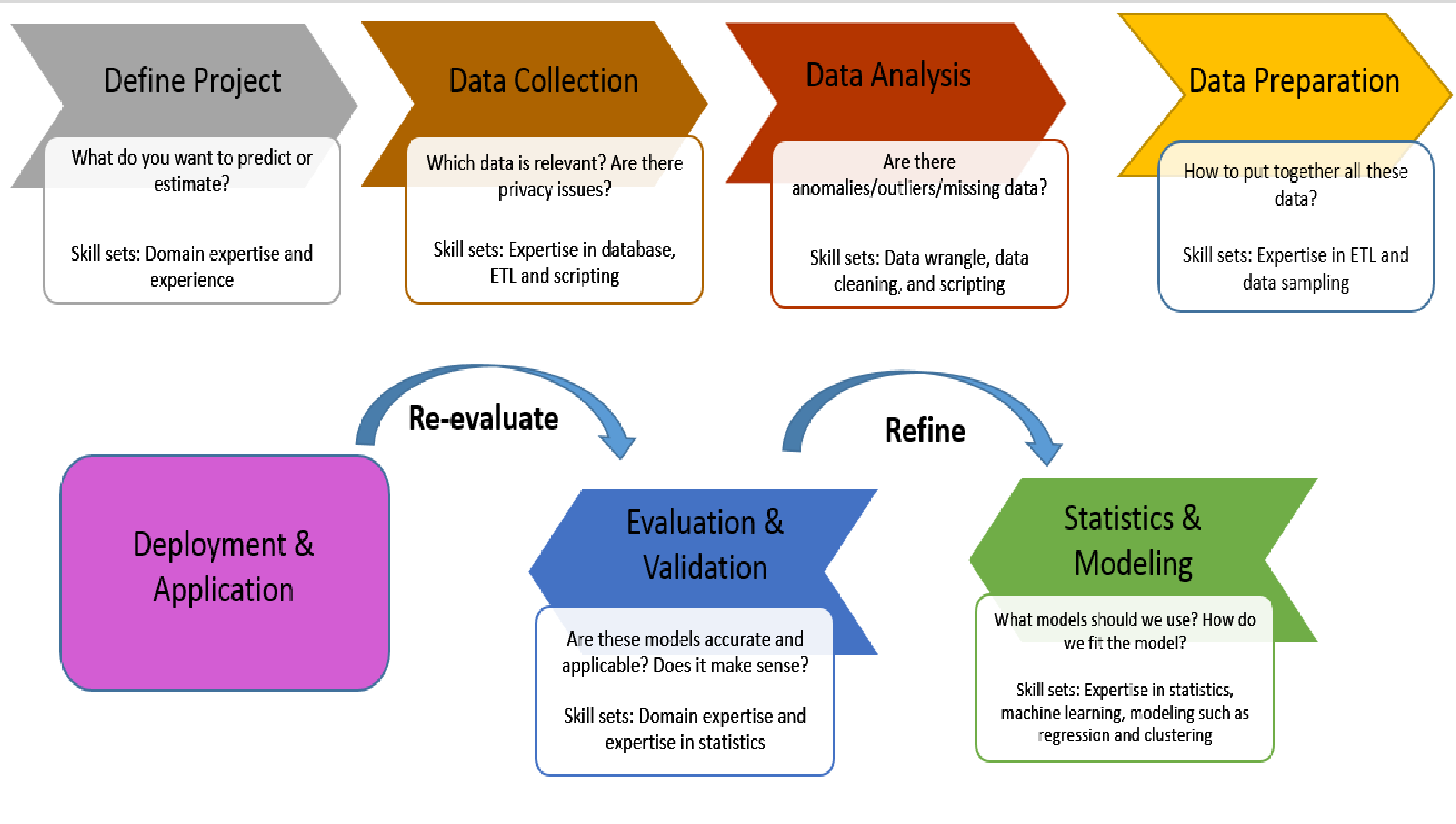


Customer_GK	claimRatio	Claims	Pred	Score
512	-0.1695	1	1	0.738574
7526	-0.0577	1	1	0.738711
8419	-0.1388	1	1	0.738932
3334	-0.1160	1	1	0.738981
7344	-0.1584	1	1	0.739097
1697	-0.1236	0	1	0.73911
4516	-0.1213	1	1	0.739125
6229	-0.0329	0	1	0.73917
2945	-0.1755	1	1	0.739275
6751	0.0296	1	1	0.739312
7480	-0.1479	1	1	0.73934
5370	-0.1467	1	1	0.73936
6056	-0.1716	1	1	0.739396
8934	-0.1627	1	1	0.739458
595	-0.1880	1	1	0.739503
1816	-0.1702	1	1	0.739643
4749	-0.1496	1	1	0.739669
206	-0.1803	1	1	0.739691
8310	-0.1501	0	1	0.739732
3944	-0.1671	1	1	0.739742
4578	-0.1735	1	1	0.739906
8130	-0.0705	1	1	0.739997
2672	0.3634	1	1	0.740255
3466	-0.1827	1	1	0.740267
165	-0.1548	1	1	0.74038

Decision Tree

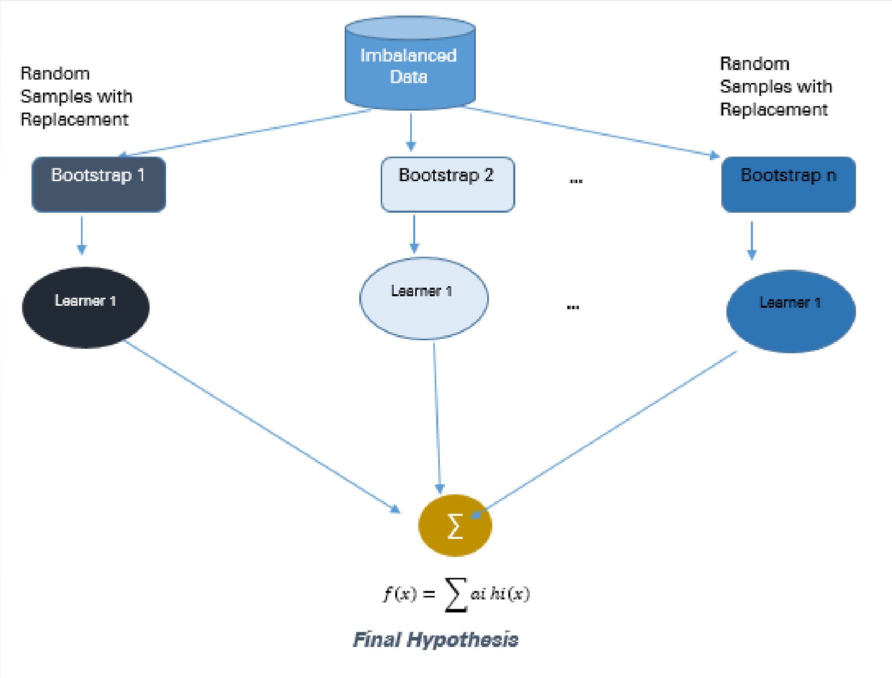


Modeling Process



Bagging

Bagging is an abbreviation of Bootstrap Aggregating. The conventional bagging algorithm involves generating ‘n’ different bootstrap training samples with replacement, training the algorithm on each bootstrapped algorithm separately and then aggregating the predictions at the end. Bagging is used for reducing Overfitting in order to create strong learners for generating accurate predictions. Unlike boosting, bagging allows replacement in the bootstrapped sample.



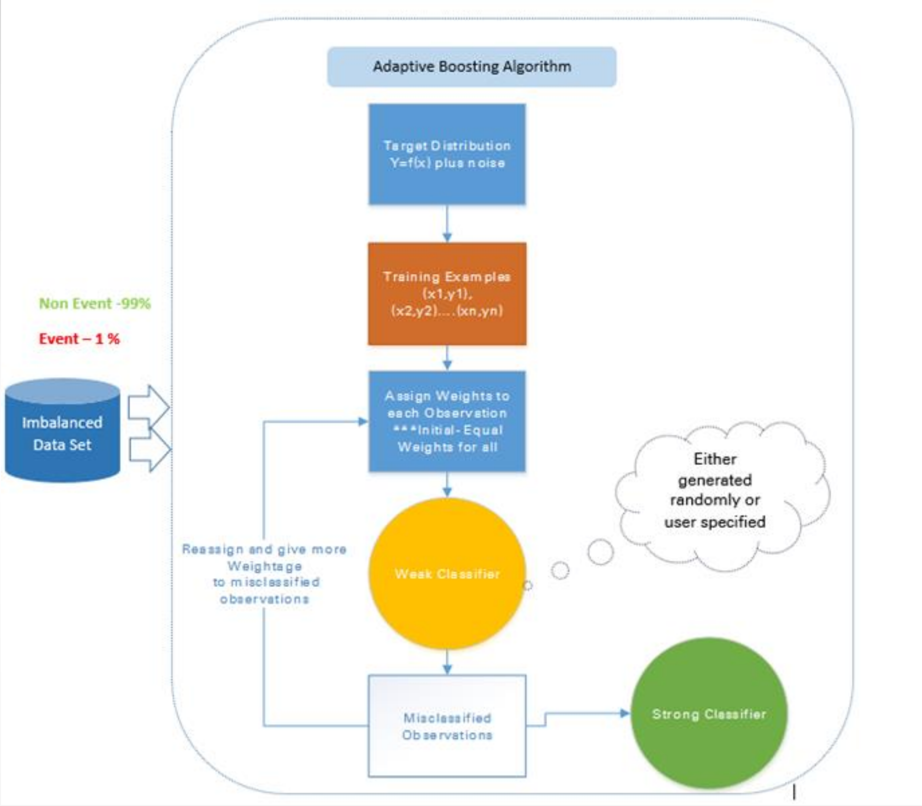
Adaptive Boosting – AdaBoost

Ada Boost is the first original boosting technique which creates a highly accurate prediction rule by combining many weak and inaccurate rules. Each classifier is serially trained with the goal of correctly classifying examples in every round that were incorrectly classified in the previous round.

For a learned classifier to make strong predictions it should follow the following three conditions:

- The rules should be simple
- Classifier should have been trained on sufficient number of training examples
- The Classifier should have low training error for the training instances

Each of the weak hypothesis has an accuracy slightly better than random guessing i.e. Error Term € (t) should be slightly more than ½-β where β >0. This is the fundamental assumption of this boosting algorithm which can produce a final hypothesis with a small error After each round, it gives more focus to examples that are harder to classify. The quantity of focus is measured by a weight, which initially is equal for all instances. After each iteration, the weights of misclassified instances are increased and the weights of correctly classified instances are decreased.



Advantages

- Very Simple to implement
- Good generalization- suited for any kind of classification problem
- Not prone to overfitting

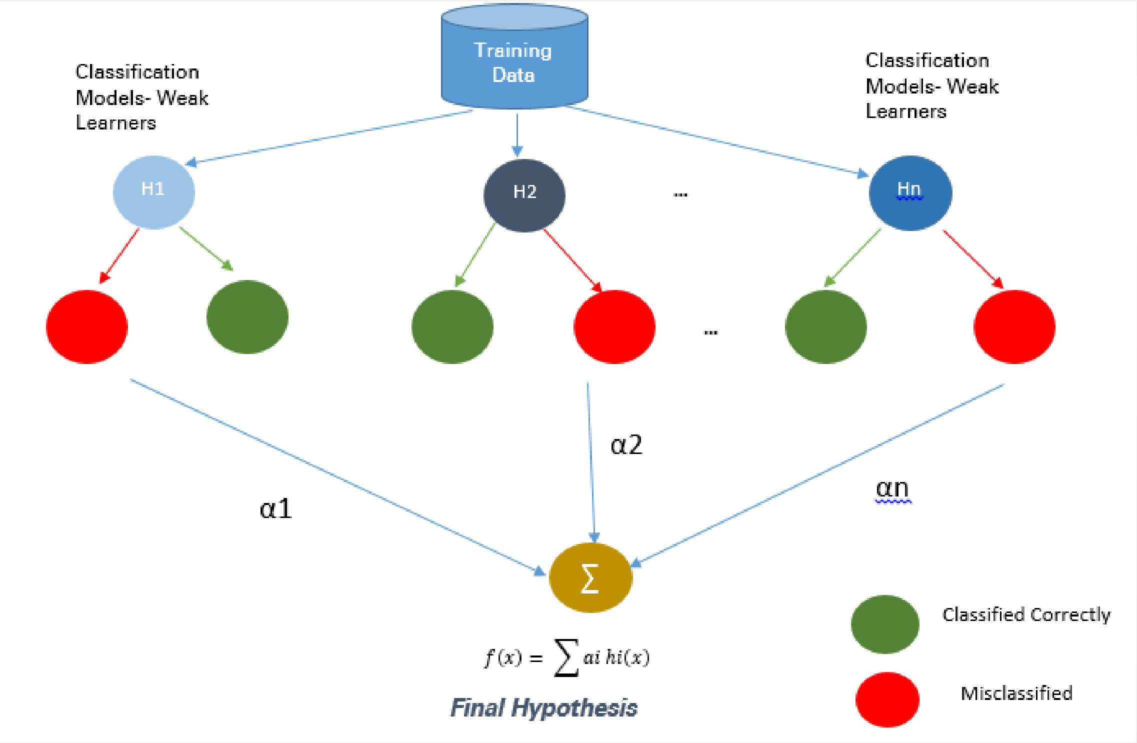
Disadvantages

- Sensitive to noisy data and outliers

Boosting-based

Boosting is an ensemble technique to combine weak learners to create a strong learner that can make accurate predictions. Boosting starts out with a base classifier / weak classifier that is prepared on the training data. The base learners / Classifiers are weak learners i.e. the prediction accuracy is only slightly better than average. A classifier learning algorithm is said to be weak when small changes in data induce big changes in the classification model.

In the next iteration, the new classifier focuses on or places more weight to those cases which were incorrectly classified in the last round.



Gradient Tree Boosting

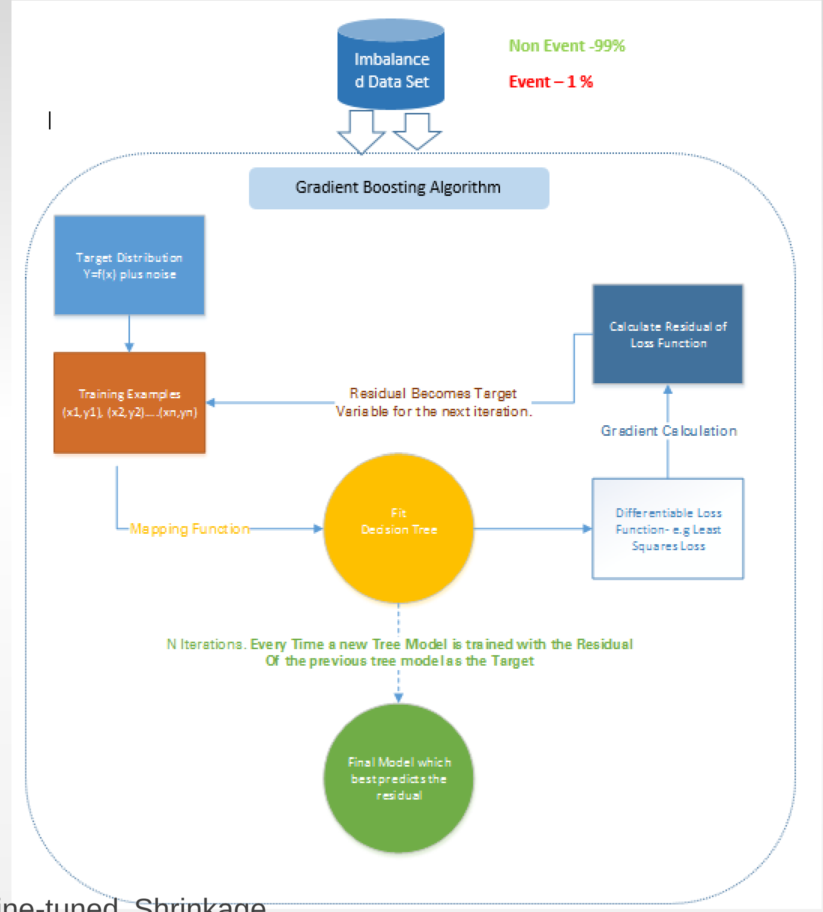
In Gradient Boosting many models are trained sequentially. It is a numerical optimization algorithm where each model minimizes the loss function, y = ax+b+e, using the Gradient Descent Method.

Decision Trees are used as weak learners in Gradient Boosting.

While both Adaboost and Gradient Boosting work on weak learners / classifiers. And try to boost them into a strong learner, there are some fundamental differences in the two methodologies. Adaboost either requires the users to specify a set of weak learners or randomly generates the weak learners before the actual learning process. The weight of each learner is adjusted at every step depending on whether it predicts a sample correctly.

On the other hand, Gradient Boosting builds the first learner on the training dataset to predict the samples, calculates the loss (Difference between real value and output of the first learner). And use this loss to build an improved learner in the second stage.

At every step, the residual of the loss function is calculated using the Gradient Descent Method and the new residual becomes a target variable for the subsequent iteration.



Disadvantages

- Gradient Boosted trees are harder to fit than random forests
- Gradient Boosting Algorithms generally have 3 parameters which can be fine-tuned, Shrinkage parameter, depth of the tree, the number of trees. Proper training of each of these parameters is needed for a good fit. If parameters are not tuned correctly it may result in over-fitting.

XGBoost

XGBoost (Extreme Gradient Boosting) is an advanced and more efficient implementation of Gradient Boosting Algorithm discussed in the previous section.

Advantages over Other Boosting Techniques

- It is 10 times faster than the normal Gradient Boosting as it implements parallel processing. It is highly flexible as users can define custom optimization objectives and evaluation criteria, has an inbuilt mechanism to handle missing values.
- Unlike gradient boosting which stops splitting a node as soon as it encounters a negative loss, XG Boost splits up to the maximum depth specified and prunes the tree backward and removes splits beyond which there is an only negative loss.

Extreme gradient boosting can be done using the XGBoost package in R and Python