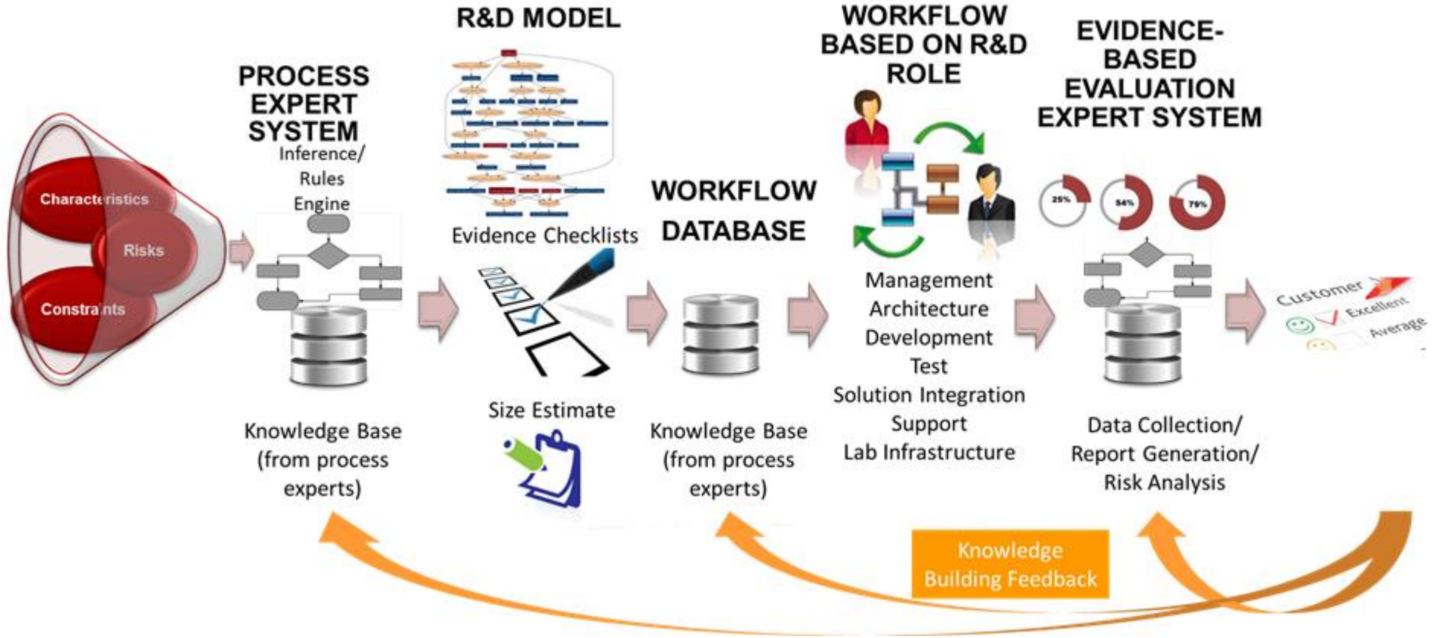
Intelligent Management Prototype
Phase V

Risk-based Task Recommendation via Machine Learning Techniques

Scope & System Architecture



- Adaptive, Evidence-based Process using machine learning
- Workflow Determination & Automation
- Automated Quality Assurance
- Predictable Results – Cost, Progress & Quality (automated tracking of progress)

Accomplishment to Date

- Workflow/Evidence Determination & Risk Analysis
- Effort Estimates & Evidence Analysis
- Automated Quality Assessment Using Data Normalization Architecture
- Phase IV
    - AI/Machine learning (ML) techniques and principles to demonstrate automatic revisions to the development process, using Incremental Commitment Spiral Model (ICSM) and automated Quality Assurance.
    - The TR4 milestone is the use case and data from the following sources is used for analysis/learning: Development Risk drivers, Defect Prediction factors, Huawei's Quality Management database, and static code analysis.
- **Phase V**
    - **Integrate risk prediction model with advanced tollgates of system prototype.**
    - **Explore open source project data to improve risk prediction model.**

Overview

- **Operational Concept**
- **Potential Data Sources**
- **Learning Risk Prediction Model**
- **Intelligent Risk Mitigation**
- **Demo**
- **Conclusions**
- **Next Steps**

Artificial Intelligence in Software Engineering

The Solutions:

- Industry (Companies/Products/Platforms B2B Ready):

  Code Construction / Configuration,

  Quality Management / Testing,

  Maintenance;

- Academic (Researches):

  Requirements, UCDD NARCIA RETA(RUBRIC)

  Code Construction / Configuration; DeepCoder FlashMeta RobustFill

Potential Fields:

Design, **Project Management**;

Source: Valentin Grigoryevskiy, AI for Software Engineering - Industry Landscape (18/Oct/2017)

Operational Concept

At each anchor point / milestone within the software development lifecycle,

The system

quantifies risks based on a set of criteria tracked from existing data source,

then **redirects to different tasks** based on the potential risks.

...

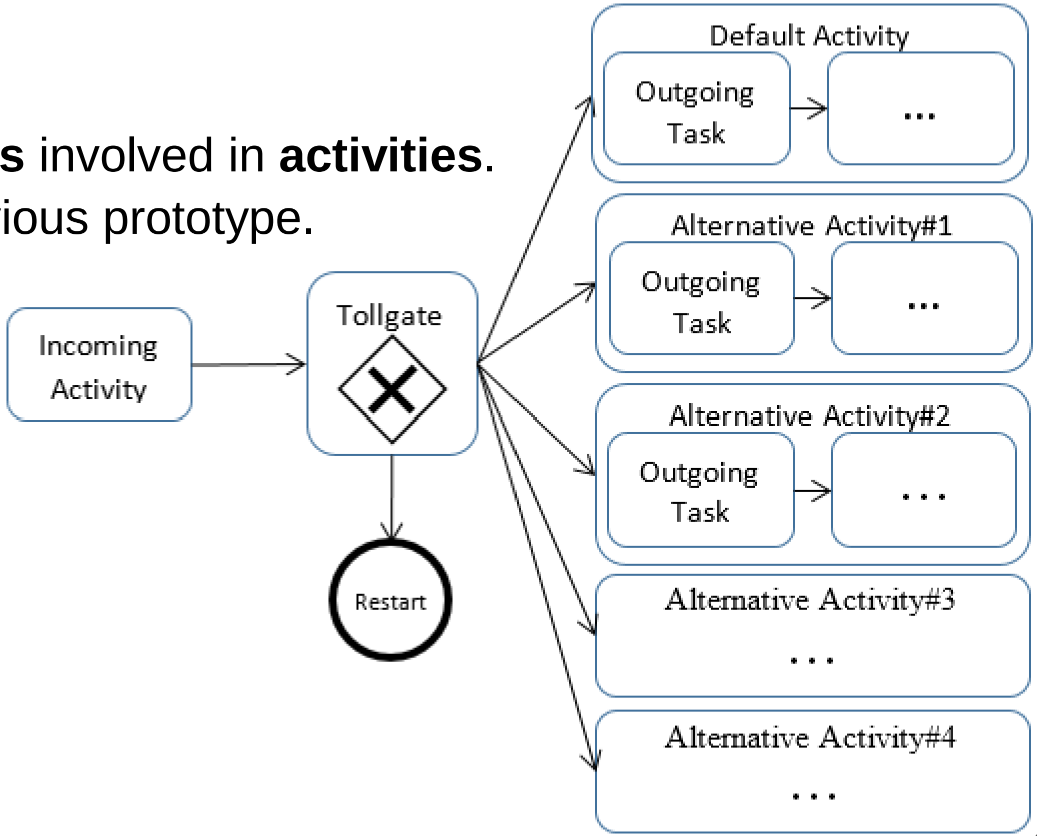
In order to present a concrete idea, current anchor point / milestone is set to

「**TR4**」

Terms

Workflow

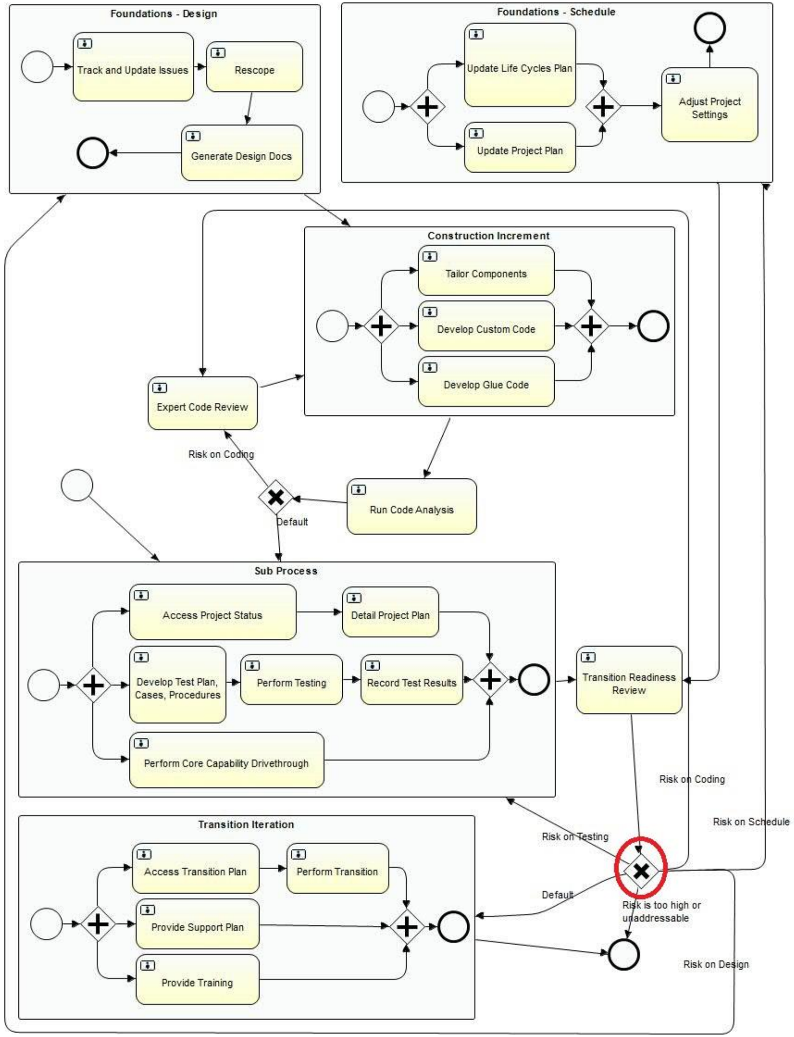
- Describes the **roles** and **tasks** involved in **activities**.
- Modeled by **BPMN2.0** in previous prototype.



Tollgate connects

- ***Incoming activities***
  which consists of a series of tasks.
- ***Outgoing activities***
  which consists of a series of tasks.

Examples

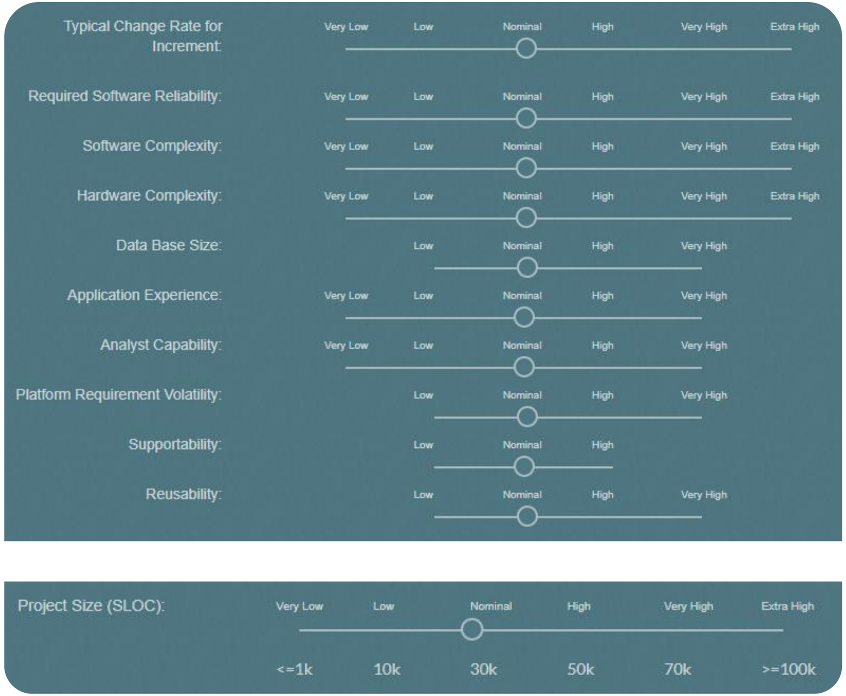


Foundations - Design
Track and Update Issues
Rescope
Generate Design Docs
Foundations - Schedule
Update Life Cycles Plan
Update Project Plan
Adjust Project Settings
Construction Increment
Tailor Components
Develop Custom Code
Develop Glue Code
Expert Code Review
Risk on Coding
Default
Run Code Analysis
Sub Process
Access Project Status
Detail Project Plan
Develop Test Plan, Cases, Procedures
Perform Testing
Record Test Results
Perform Core Capability Drivethrough
Transition Readiness Review
Risk on Coding
Risk on Schedule
Risk on Testing
Default
Risk is too high or unaddressable
Risk on Design
Transition Iteration
Access Transition Plan
Perform Transition
Provide Support Plan
Provide Training

Potential

Data Sources

Development Risk Drivers from Project Initialization



- Expressive factors while initializing a project
  - **22 factors**: reflects the product, process, personnel, and platform of the project.
  - Have been integrated into the current prototype

Estimated	Effort (Person-Months)	Schedule Duration (Months)	Productivity (KSLOC per Month)	Staff (Number of FTEs)
Optimistic:	340.93	21.77	5.05	15.66
Most Likely:	426.16	23.31	4.72	18.29
Pessimistic:	532.70	24.95	4.41	21.35

Based on Dr. Barry Boehm's study of Constructive Cost Model (COCOMO)

Defect Prediction Factors

- **3 Baseline Defect Rates** in addition to the project initialization settings
  - **Automated Analysis**
  - **Peer Reviews**
  - **Level of Testing Sophistication**

Defect Introduction Drivers

PREC N | RESL N | TEAM N | PMAT N | FLEX N
RELY N | DATA N | DOCU N | CPLX N | RUSE N
TIME N | STOR N | PVOL N
ACAP N | AEPX N | PCAP N | PEXP N | LTEX N | PCON N
TOOL N | SCED N | SITE N

Defect Removal Profiles

Automated Analysis N | Peer Reviews N | Execution Testing and Tools N

	Requirements	Design	Code	Total
Number of Defects Introduced	60	120	180	360
Number of Defects Removed	40.56	84.295	148.55	273.406
Number of Defects Remaining	19.44	35.705	31.45	86.5944

Based on Dr. Raymond Madachy’s study of Constructive Quality Model (COQUALMO)

Defect Removal Rating Scales

	Very Low	Low	Nominal	High	Very High	Extra High
Automated Analysis	Simple compiler syntax checking	Basic compiler capabilities	Compiler extension Basic req. and design consistency	Intermediate-level module Simple req./design	More elaborate req./design Basic dist-processing	Formalized specification, verification. Advanced dist-processing
Peer Reviews	No peer review	Ad-hoc informal walk-through	Well-defined preparation, review, minimal follow-up	Formal review roles and Well-trained people and basic checklist	Root cause analysis, formal follow Using historical data	Extensive review checklist Statistical control
Level of Testing Sophistication	No testing	Ad-hoc test and debug	Basic test Test criteria based on checklist	Well-defined test seq. and basic test coverage tool system	More advance test tools, preparation. Dist-monitoring	Highly advanced tools, model-based test

Code Repository Analysis (SQUAAD)

Understanding Software Quality Evolution Using SQUAAD

- Two approaches
    - Absolute value of quality attributes after each commit
    - Impact of developers on quality attributes at each commit
- Quality Attributes
    - A collection of 50+ metrics
    - Example
        - M1: Code Smells
        - M2: Security Vulnerabilities
- How can we use this data?
    - If the number of code smells (M1) exceeds a certain amount maintenance tasks should get higher priority.

If the number of security vulnerabilities (M2) has increased in the last commit maintenance tasks should get higher priority.

SQUAAD is a tool from USC-CSSE to evaluate code evolution for open source projects (by Pooyan Behnamghader).

List of All Metrics

Tool: SonarQube

Classes
Comment_lines_density
Vulnerabilities
Lines
Ncloc
Complexity
Security_rating
Major_violations
Duplicated_blocks
Code_smells
File_complexity
Functions
Duplicated_files
Duplicated_lines_density
Reliability_rating
Critical_violations
Violations
Statements
Blocker_violations
Reliability_remediation_effort
Duplicated_lines
Bugs
Security_remediation_effort
Directories
Info_violations
Sqale_index
Sqale_debt_ratio
Minor_violations
Files
Sqale_rating

Tool: PMD

Basic
Emptycode
Clone implementation
Comments
Codesize
String and stringbuffer
Naming
Strict exceptions
Optimization
Design
Security code guidelines
Braces
Type resolution
Coupling
Finalizer
Import statements
Unused code
Unnecessary

Tool: FindBugs

Security
I18n
Mt_correctness
Style
Experimental
Correctness
Malicious_code
Bad_practice
Performance

Code Smells (SQUAAD)

A maintainability-related issue in the code.

- Leaving it as-is means that at best maintainers will have a harder time than they should making changes to the code.
- At worst, they'll be so confused by the state of the code that they'll introduce additional errors as they make changes.

Example:

- Child class fields should not shadow parent class fields.
  - Having a variable with the same name in two unrelated classes is fine, but do the same thing within a class hierarchy and you'll get confusion at best, chaos at worst.

NONCOMPLIANT CODE EXAMPLE

```
public class Fruit {
  protected Season ripe;
  protected Color flesh;
  // ...
}
public class Raspberry extends Fruit {
  private boolean ripe;  // Noncompliant
  private static Color FLESH; // Noncompliant
}
```

COMPLIANT SOLUTION

```
public class Fruit {
  protected Season ripe;
  protected Color flesh;
  // ...
}
public class Raspberry extends Fruit {
  private boolean ripened;
  private static Color FLESH_COLOR;
}
```

Security Vulnerabilities (SQUAAD)

A security-related issue which represents a potential backdoor for attackers.

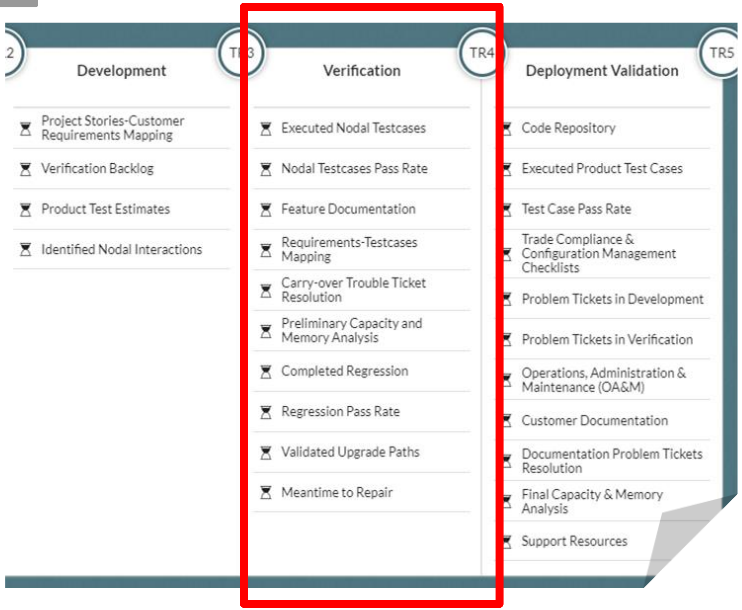
Example:

- SQL binding mechanisms should be used
  - Applications that execute SQL commands should neutralize any externally-provided values used in those commands.
  - Failure to do so could allow an attacker to include input that changes the query so that unintended commands are executed, or sensitive data is exposed.

```
public User getUser(Connection con, String user) throws SQLException {
  Statement stmt1 = null;
  Statement stmt2 = null;
  PreparedStatement pstmt;
  try {
    stmt1 = con.createStatement();
    ResultSet rs1 = stmt1.executeQuery("GETDATE()"); // Compliant; parameters not used here
    stmt2 = con.createStatement();
    ResultSet rs2 = stmt2.executeQuery("select FNAME, LNAME, SSN " +
                 "from USERS where UNAME=" + user);  // Noncompliant; parameter concatenated directly into query
    pstmt = con.prepareStatement("select FNAME, LNAME, SSN " +
                 "from USERS where UNAME=" + user);  // Noncompliant; parameter concatenated directly into query
    ResultSet rs3 = pstmt.executeQuery();
    //...
}
public User getUserHibernate(org.hibernate.Session session, String userInput) {
  org.hibernate.Query query = session.createQuery(  // Compliant
         "FROM students where fname = " + userInput);  // Noncompliant; parameter binding should be used instead
  // ...
}
```

Evidence Criteria

- **Success Criteria** of the Evidence in 「TR4」



TR4/DCR2 QA Inputs

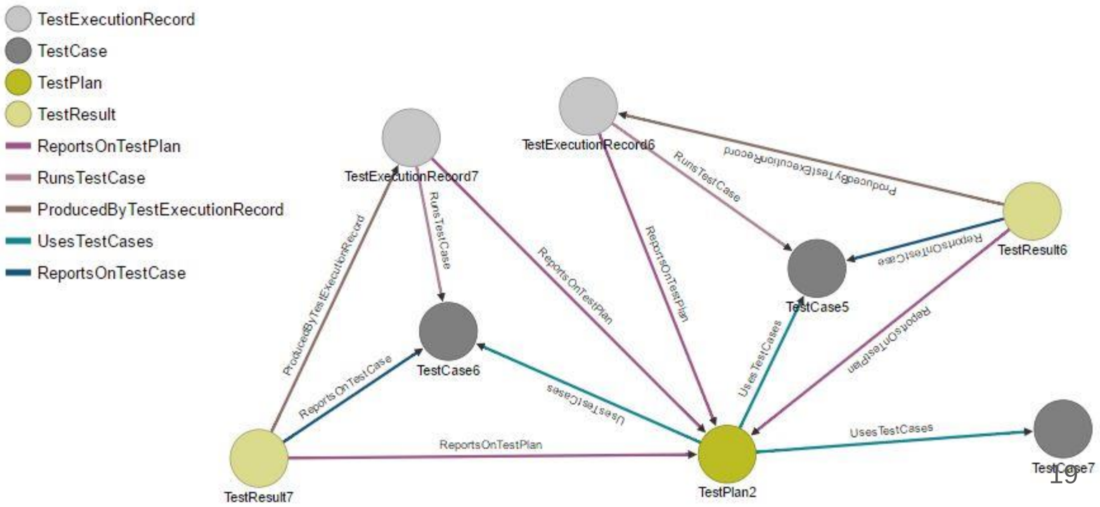
Key Performance Indicator	Threshold
Nodal Testcases executed	99%
Nodal Testcases Pass Rate	95%
Feature Documentation complete including the actual implementation	95%
Requirements Mapping to test cases captured in Repository	90%
Trouble Tickets carried over from older releases resolved	100%
Preliminary capacity and memory analysis	100%
Regression completed	100%
Regression pass rate	95%
Upgrade paths validated	100%
Problem tracking work-on-hand	<1.5 weeks

Quality Management Database
From Huawei’s DNA

5 Factors can be used in the current model

- **FCR (feature completion rate)**
- **CD (number of code defects)**
- **ISS (number of reported issues at the feature level)**
- **ISRR (issue removal rate)**
- **DRR (defect removal rate)**

TR4



Issue Resolution Analysis

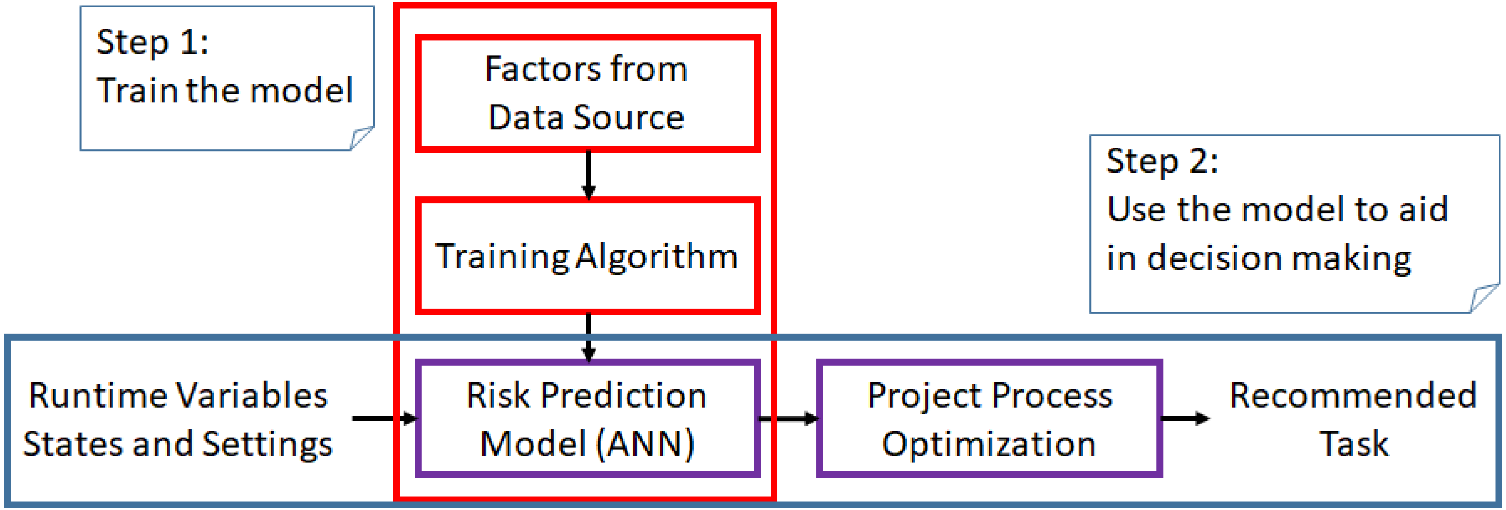
Understanding Issue Resolution Status

- Collect information from JIRA and categorized by milestone
  - Get to know which of the still-open issues at the end of phase 1 were still open in later phases, as they would be still accumulating technical debt.
  - Get to know if high-severity issues were given higher priorities for closure.
- Factors
  - ISS_Num_Resolved: # of resolved issues
  - ISS_Num_Unresolved: # of unresolved issues
  - Personnel: # of contributors
  - Estimated_Effort: Estimated total cost
  - Accu_Trival: Accumulated unsolved issues at Trival Level
  - Accu_Minor: Accumulated unsolved issues at Minor Level
  - ...

Data Source Categories

- Project Settings
  - Development risk drivers
  - Defect prediction factors
- Runtime States
  - Issues on Design, Implementation, Testing and Technical Debts
    - Code Repository Analysis
    - Issue Resolution Analysis
    - Quality Management Database
- All those required data are supposed to be automatically extracted from different data sources during the runtime.

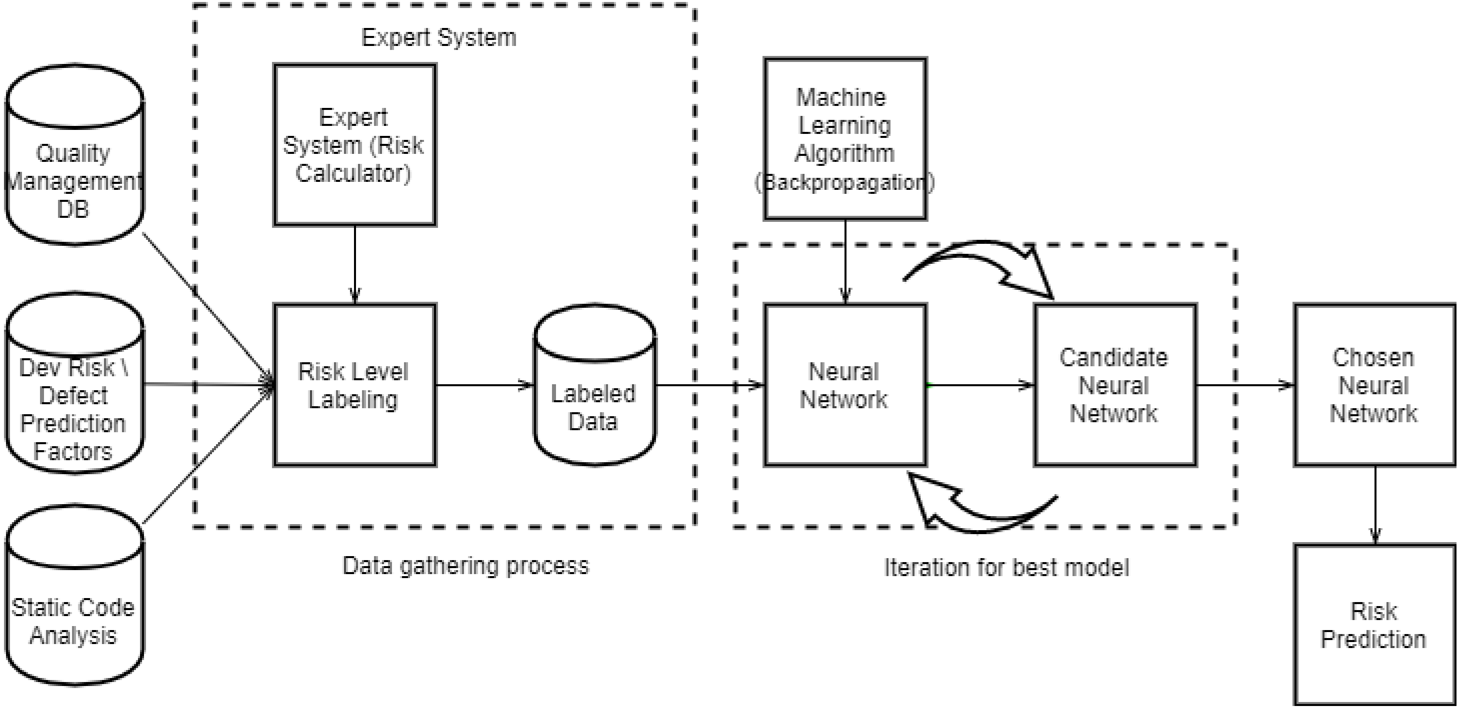
Integration with Machine Learning



Learning

Risk Prediction Model

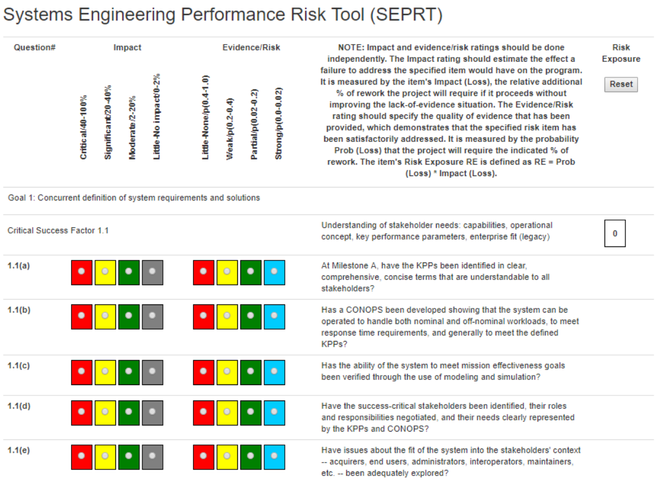
The Model Building Process



The Data Collection Process

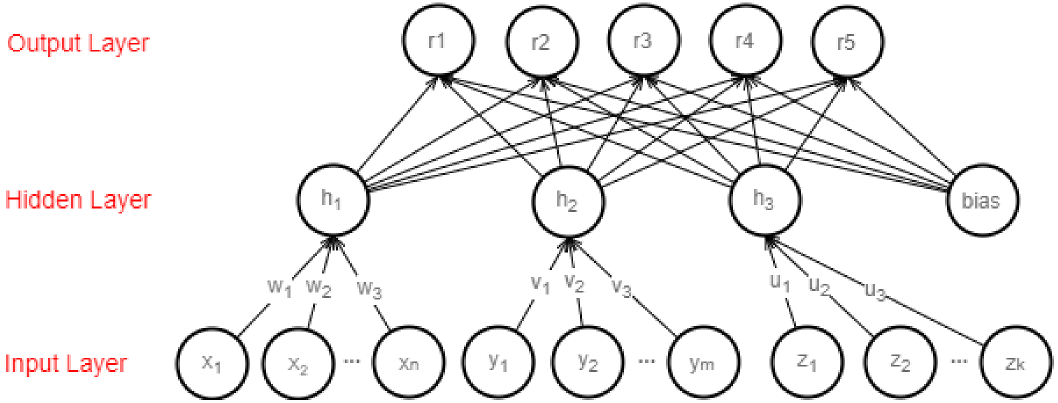
- **Collecting data points by the inputs of the proposed model**
  - Group X: The results of automatic code static analysis,
  - Group Y: The inputs from project settings when initializing a project in the system.
  - Group Z: The data entries from the quality management database (open source project data as supplement).
- **When put into production, the data assumed by the model will automatically generated/input from the user interface of the expert system. No requirements of extra effort of manually inputting the data.**

- **Labeling data points.**
  - Experienced project Managers label the current status of a project into different levels of risk.
  - The Risk calculator (developed in phase II) provides the guideline for the experts.
  - Other Domain knowledge and criteria may also apply, if the expert is confident, for example, by the criteria of being over budget or schedule



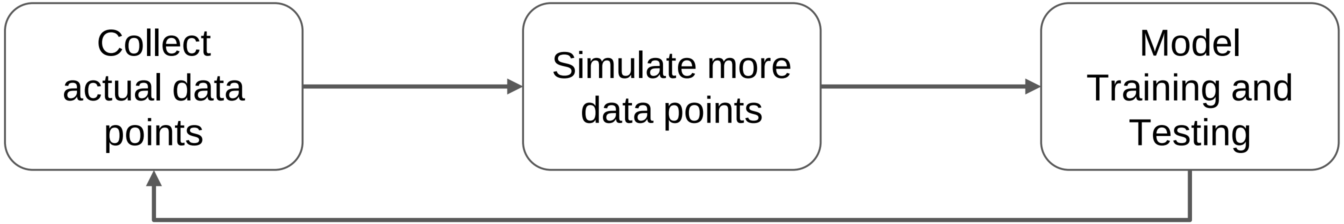
Risk Prediction Neural Network

- Classify the current status of a project into five levels of risk based on the defined inputs.
- Multilayer Perceptron with 3-layer architecture: Input layer, Hidden layer, and Output Layer.
  - The neurons in the hidden layer are the general representations of the inputs.
  - May generalise more hidden layers as needed.



Current Implementation of Risk Prediction Model

1. Actual data from Open Source Projects and USC 577 software engineering courses.
   a. Open source projects: ACUMOS - ATT, OPNFV - ATT, APOLLO - Baidu, **Carbon Data-Huawei, OpenSDS-Huawei**, HYPERLEDGER - IBM.
   b. USC master-level engineering sources: 22 projects.
2. Simulated data for model training and testing.
   a. 500 data points based on covariance matrices of the empirical data.
3. Using Neuralnet Package in R to train the model.
   a. Demo API is available at our EC2 server



Risk Estimation Model based on **USC-CSSE Projects**

Data Source:

- 25 Master-level software engineering projects from 2015-2017. Formal method (ICSM) was applied. Those project involved real world clients and delivered complete products.

Measures:

- Risk drivers describe the relevant characteristics of a project in terms of personnel, platform, process, and product aspects.
- They describe the potential risks from products and platforms.
- They describe ability of the people and the process to undertake the risks.
- A combination of those factors describes the general bounds of the levels of risks.

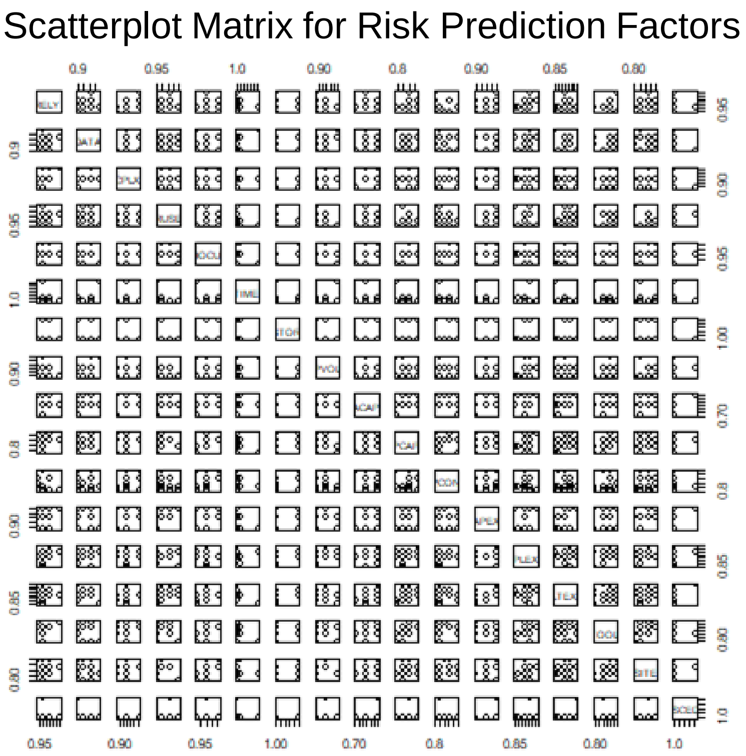
Empirically collected data - Risk Drivers

Development Risk Drivers

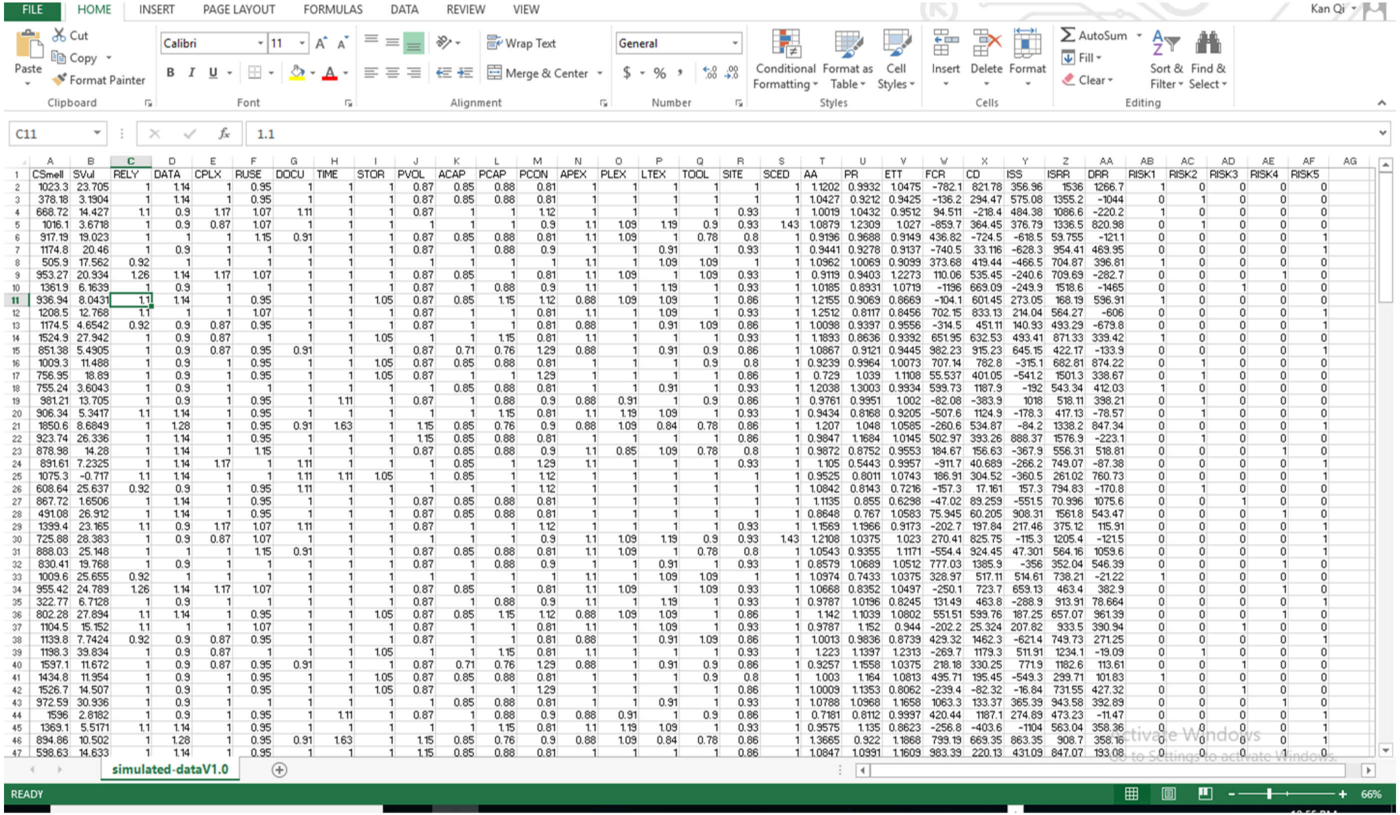
1	Team. NO	Semester	PREC	FLEX	RESL	TEAM	PMAT	RELY	DATA	CPLX	RUSE	DOCU	TIME	STOR	PVOL	ACAP	PCAP	PCON	APEX	PLEX	LTEX	TOOL	SITE	SCED
2	1	fall2015/s	L	H	H	VH	N	N	H	N	L	N	N	N	L	H	H	VH	N	N	N	N	N	N
3	2	fall2015/s	L	H	H	VH	N	N	H	N	L	N	N	N	L	H	H	VH	N	N	N	N	N	N
4	3	fall2015	L	N	N	H	N	H	L	H	H	H	N	N	L	N	N	L	N	N	N	N	H	N
5	4	fall2015	H	H	N	H	N	N	L	L	H	N	N	N	N	N	N	H	L	L	VL	H	H	VL
6	5	fall2015	H	H	L	H	L	N	N	N	VH	L	N	N	L	H	H	VH	L	L	N	VH	EXH	N
7	6	fall2015	N	N	H	N	N	N	L	N	N	N	N	N	L	N	H	H	N	N	H	N	H	N
8	7	fall2015	L	L	L	L	N	L	N	N	N	N	N	N	N	N	N	N	L	N	L	L	N	N
9	2	fall2014	L	H	N	H	N	VH	H	H	H	N	N	N	L	H	N	VH	L	L	N	L	H	N
10	5	fall2014	N	N	H	H	N	N	L	N	N	N	N	N	L	N	H	H	L	N	VL	N	H	N
11	6	fall2014	H	H	H	VH	N	H	H	N	L	N	N	H	L	H	L	L	H	L	L	N	VH	N
12	8	fall2014	N	H	N	N	H	H	N	N	H	N	N	N	L	N	N	VH	L	N	L	N	H	N
13	9	fall2014	H	L	N	N	N	L	L	L	L	N	N	N	L	N	N	VH	H	N	H	L	VH	N
14	11	fall2014	H	H	N	H	N	N	L	L	N	N	N	H	N	N	L	VH	L	N	N	N	H	N
15	13	fall2014	N	N	L	H	N	N	L	L	L	L	N	N	L	VH	VH	VL	H	N	H	H	VH	N
16	14	fall2014	L	H	H	H	L	N	L	N	L	N	N	H	L	H	H	VH	N	N	N	H	EXH	N
17	15	fall2014	N	N	N	H	N	N	L	N	L	N	N	H	L	N	N	VL	N	N	N	N	VH	N
18	1	fall2014/s	H	H	N	H	N	N	L	N	N	N	N	N	N	H	H	VH	N	N	H	N	H	N
19	7	fall2014/s	N	VH	N	H	N	N	L	N	L	N	H	N	L	N	H	H	H	H	N	H	VH	N
20	4	fall2014/s	L	N	L	VH	N	H	H	N	L	N	N	N	N	N	L	VH	L	VL	L	N	H	N
21	10	fall2014/s	H	VH	H	VH	N	N	VH	N	L	L	EXH	N	H	H	VH	H	H	L	VH	VH	VH	N
22	12	fall2014/s	N	N	H	H	N	N	H	N	L	N	N	N	H	H	H	VH	N	N	N	N	VH	N
23	3	fall2014/s	H	H	L	H	L	N	H	N	VH	N	N	N	L	H	H	H	L	VH	L	VH	EXH	N
24	1	spring201	N	N	H	H	N	N	H	H	N	H	N	N	N	H	N	VL	L	N	N	N	H	N
25	3	spring201	N	N	L	N	N	H	H	N	N	H	H	H	N	H	N	L	N	N	N	N	N	N

COCOMOData

Scatter Matrix for Data Simulation

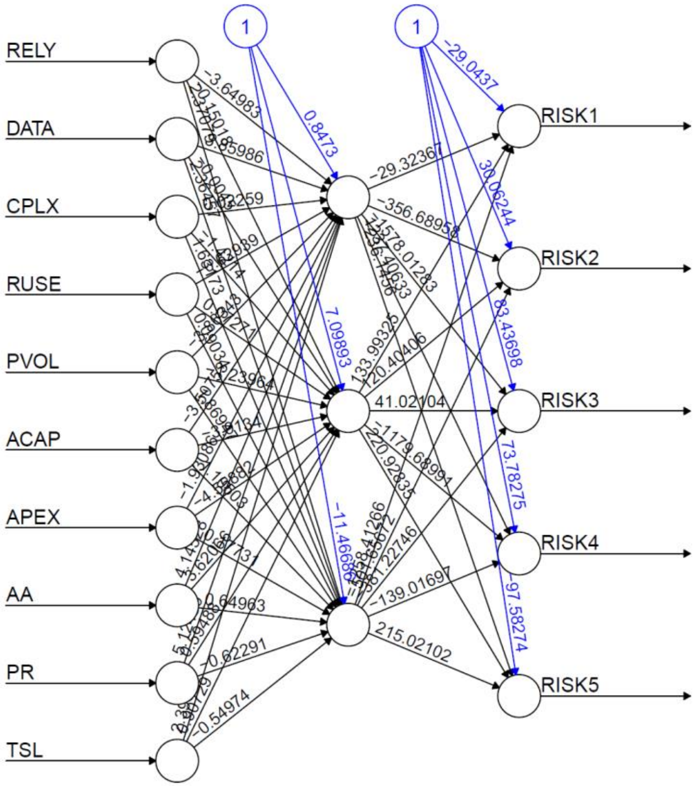


Example of Simulated Data



CSmell	SVul	RELY	DATA	CPLX	RUSE	DOCU	TIME	STOR	PVOL	ACAP	PCAP	PCON	APEX	PLEX	LTEX	TOOL	SITE	SCED	AA	PR	ETT	FCR	CD	ISS	ISRR	DRR	RISK1	RISK2	RISK3	RISK4	RISK5
1023.3	23.705	1	1.14	1	0.95	1	1	1	0.87	0.85	0.88	0.81	1	1	1	1	1	1	1.1202	0.9932	1.0475	-782.1	821.78	356.96	1536	1266.7	1	0	0	0	0
378.18	3.1904	1	1.14	1	0.95	1	1	1	0.87	0.85	0.88	0.81	1	1	1	1	1	1	1.0427	0.9212	0.9425	-136.2	294.47	575.08	1355.2	-1044	0	1	0	0	0
668.72	14.427	1.1	0.9	1.17	1.07	1.11	1	1	0.87	1	1	1.12	1	1	1	1	0.93	1	1.0019	1.0432	0.9512	94.511	-218.4	484.38	1086.6	-220.2	1	0	0	0	0
1016.1	3.6718	1	0.9	0.87	1.07	1	1	1	1	1	1	0.9	1.1	1.09	1.19	0.9	0.93	1.43	1.0879	1.2309	1.027	-859.7	364.45	376.79	1336.5	820.98	0	1	0	0	0
917.19	19.023	1	1	1	1.15	0.91	1	1	0.87	0.85	0.88	0.81	1.1	1.09	1	0.78	0.8	1	0.9196	0.9688	0.9149	436.82	-724.5	-618.5	59.755	-121.1	0	0	0	0	1
1174.8	20.46	1	0.9	1	1	1	1	1	0.87	1	0.88	0.9	1	1	0.91	1	0.93	1	0.9441	0.9278	0.9137	-740.5	33.116	-628.3	954.41	469.95	0	0	0	0	1
505.9	17.562	0.92	1	1	1	1	1	1	1	1	1	1	1.1	1	1.09	1.09	1	1	1.0962	1.0069	0.9099	373.68	419.44	-466.5	704.87	396.81	1	0	0	0	0
953.27	20.934	1.26	1.14	1.17	1.07	1	1	1	0.87	0.85	1	0.81	1.1	1.09	1	1.09	0.93	1	0.9119	0.9403	1.2273	110.06	535.45	-240.6	709.69	-282.7	0	0	0	1	0
1361.9	6.1639	1	0.9	1	1	1	1	1	0.87	1	0.88	0.9	1.1	1	1.19	1	0.93	1	1.0185	0.8931	1.0719	-1196	669.09	-249.9	1518.6	-1465	0	0	1	0	0
936.94	8.0431	1.1	1.14	1	0.95	1	1	1.05	0.87	0.85	1.15	1.12	0.88	1.09	1.09	1	0.86	1	1.2155	0.9069	0.8669	-104.1	601.45	273.05	168.19	596.91	1	0	0	0	0
1208.5	12.768	1.1	1	1	1.07	1	1	1	0.87	1	1	0.81	1.1	1	1.09	1	0.93	1	1.2512	0.8117	0.8456	702.15	833.13	214.04	564.27	-606	0	0	0	0	1
1174.5	4.6542	0.92	0.9	0.87	0.95	1	1	1	0.87	1	1	0.81	0.88	1	0.91	1.09	0.86	1	1.0098	0.9397	0.9556	-314.5	451.11	140.93	493.29	-679.8	0	0	0	0	1
1524.9	27.942	1	0.9	0.87	1	1	1	1.05	1	1	1.15	0.81	1.1	1	1	1	0.93	1	1.1893	0.8636	0.9392	651.95	632.53	493.41	871.33	339.42	1	0	0	0	0
851.38	5.4905	1	0.9	0.87	0.95	0.91	1	1	0.87	0.71	0.76	1.29	0.88	1	0.91	0.9	0.86	1	1.0867	0.9121	0.9445	982.23	915.23	645.15	422.17	-133.9	0	0	0	0	1
1009.3	11.488	1	0.9	1	0.95	1	1	1.05	0.87	0.85	0.88	0.81	1	1	1	0.9	0.8	1	0.9239	0.9964	1.0073	707.14	782.8	-315.1	682.81	874.22	0	1	0	0	0
756.95	18.89	1	0.9	1	0.95	1	1	1.05	0.87	1	1	1.29	1	1	1	1	0.86	1	0.729	1.039	1.1108	55.537	401.05	-541.2	1501.3	338.67	0	1	0	0	0
755.24	3.6043	1	0.9	1	1	1	1	1	1	0.85	0.88	0.81	1	1	0.91	1	0.93	1	1.2038	1.3003	0.9934	599.73	1187.9	-192	543.34	412.03	1	0	0	0	0
981.21	13.705	1	0.9	1	0.95	1	1.11	1	0.87	1	0.88	0.9	0.88	0.91	1	0.9	0.86	1	0.9761	0.9951	1.002	-82.08	-383.9	1018	518.11	398.21	0	1	0	0	0
906.34	5.3417	1.1	1.14	1	0.95	1	1	1	1	1	1.15	0.81	1.1	1.19	1.09	1	0.93	1	0.9434	0.8168	0.9205	-507.6	1124.9	-178.3	417.13	-78.57	0	1	0	0	0
1850.6	8.6849	1	1.28	1	0.95	0.91	1.63	1	1.15	0.85	0.76	0.9	0.88	1.09	0.84	0.78	0.86	1	1.207	1.048	1.0585	-260.6	534.87	-84.2	1338.2	847.34	0	0	0	0	1
923.74	26.336	1	1.14	1	0.95	1	1	1	1.15	0.85	0.88	0.81	1	1	1	1	0.86	1	0.9847	1.1684	1.0145	502.97	393.26	888.37	1576.9	-223.1	0	1	0	0	0
878.98	14.28	1	1.14	1	1.15	1	1	1	0.87	0.85	0.88	0.9	1.1	0.85	1.09	0.78	0.8	1	0.9872	0.8752	0.9553	184.67	156.63	-367.9	556.31	518.81	0	0	0	1	0
891.61	7.2325	1	1.14	1.17	1	1.11	1	1	1	0.85	1	1.29	1.1	1	1	1	0.93	1	1.105	0.5443	0.9957	-911.7	40.689	-266.2	749.07	-87.38	0	0	0	0	1
1075.3	-0.717	1.1	1.14	1	1	1.11	1.11	1.05	1	0.85	1	1.12	1	1	1	1	1	1	0.9525	0.8011	1.0743	186.91	304.52	-360.5	261.02	760.73	0	0	0	0	1
608.64	25.637	0.92	0.9	1	0.95	1.11	1	1	1	1	1	1.12	1	1	1	1	1	1	1.0842	0.8143	0.7216	-157.3	17.161	157.3	794.83	-170.8	0	1	0	0	0
867.72	1.6506	1	1.14	1	0.95	1	1	1	0.87	0.85	0.88	0.81	1	1	1	1	1	1	1.1135	0.855	0.6298	-47.02	89.259	-551.5	70.996	1075.6	0	0	1	0	0
491.08	26.912	1	1.14	1	0.95	1	1	1	0.87	0.85	0.88	0.81	1	1	1	1	1	1	0.8648	0.767	1.0583	75.945	60.205	908.31	1561.8	543.47	0	0	0	1	0
1399.4	23.165	1.1	0.9	1.17	1.07	1.11	1	1	0.87	1	1	1.12	1	1	1	1	0.93	1	1.1569	1.1966	0.9173	-202.7	197.84	217.46	375.12	115.91	0	0	0	0	1
725.88	28.383	1	0.9	0.87	1.07	1	1	1	1	1	1	0.9	1.1	1.09	1.19	0.9	0.93	1.43	1.2108	1.0375	1.023	270.41	825.75	-115.3	1205.4	-121.5	0	0	0	0	1
888.03	25.148	1	1	1	1.15	0.91	1	1	0.87	0.85	0.88	0.81	1.1	1.09	1	0.78	0.8	1	1.0543	0.9355	1.1171	-554.4	924.45	47.301	564.16	1059.6	0	0	0	0	1
830.41	19.768	1	0.9	1	1	1	1	1	0.87	1	0.88	0.9	1	1	0.91	1	0.93	1	0.8579	1.0689	1.0512	777.03	1385.9	-356	352.04	546.39	0	0	0	1	0
1009.6	25.655	0.92	1	1	1	1	1	1	1	1	1	1	1.1	1	1.09	1.09	1	1	1.0974	0.7433	1.0375	328.97	517.11	514.61	738.21	-21.22	1	0	0	0	0
955.42	24.789	1.26	1.14	1.17	1.07	1	1	1	0.87	0.85	1	0.81	1.1	1.09	1	1.09	0.93	1	1.0668	0.8352	1.0497	-250.1	723.7	659.13	463.4	382.9	0	0	0	1	0
322.77	6.7128	1	0.9	1	1	1	1	1	0.87	1	0.88	0.9	1.1	1	1.19	1	0.93	1	0.9787	1.0196	0.8245	131.49	463.8	-288.9	913.91	78.664	0	0	0	0	1
802.28	27.894	1.1	1.14	1	0.95	1	1	1.05	0.87	0.85	1.15	1.12	0.88	1.09	1.09	1	0.86	1	1.142	1.1039	1.0802	551.51	599.76	187.25	657.07	961.39	0	0	0	1	0
1104.5	15.152	1.1	1	1	1.07	1	1	1	0.87	1	1	0.81	1.1	1	1.09	1	0.93	1	0.9787	1.152	0.944	-202.2	25.324	207.82	933.5	390.94	0	0	1	0	0
1139.8	7.7424	0.92	0.9	0.87	0.95	1	1	1	0.87	1	1	0.81	0.88	1	0.91	1.09	0.86	1	1.0013	0.9836	0.8739	429.32	1462.3	-621.4	749.73	271.25	0	0	0	0	1
1198.3	39.834	1	0.9	0.87	1	1	1	1.05	1	1	1.15	0.81	1.1	1	1	1	0.93	1	1.223	1.1397	1.2313	-269.7	1179.3	511.91	1234.1	-19.09	0	1	0	0	0
1597.1	11.672	1	0.9	0.87	0.95	0.91	1	1	0.87	0.71	0.76	1.29	0.88	1	0.91	0.9	0.86	1	0.9257	1.1558	1.0375	218.18	330.25	771.9	1182.6	113.61	0	0	1	0	0
1434.8	11.954	1	0.9	1	0.95	1	1	1.05	0.87	0.85	0.88	0.81	1	1	1	0.9	0.8	1	1.003	1.164	1.0813	495.71	195.45	-549.3	299.71	101.83	1	0	0	0	0
1526.7	14.507	1	0.9	1	0.95	1	1	1.05	0.87	1	1	1.29	1	1	1	1	0.86	1	1.0009	1.1353	0.8062	-239.4	-82.32	-16.84	731.55	427.32	0	0	1	0	0
972.59	30.936	1	0.9	1	1	1	1	1	1	0.85	0.88	0.81	1	1	0.91	1	0.93	1	1.0788	1.0968	1.1658	1063.3	133.37	365.39	943.58	392.89	0	0	0	1	0
1596	2.8182	1	0.9	1	0.95	1	1.11	1	0.87	1	0.88	0.9	0.88	0.91	1	0.9	0.86	1	0.7181	0.8112	0.9997	420.44	1187.1	274.89	473.23	-11.47	0	0	0	0	1
1369.1	5.5171	1.1	1.14	1	0.95	1	1	1	1	1	1.15	0.81	1.1	1.19	1.09	1	0.93	1	0.9575	1.135	0.8623	-256.8	-403.6	-1104	563.04	358.36	0	0	0	0	1
894.86	10.502	1	1.28	1	0.95	0.91	1.63	1	1.15	0.85	0.76	0.9	0.88	1.09	0.84	0.78	0.86	1	1.3665	0.922	1.1868	799.19	669.35	863.35	908.7	358.16	0	0	0	0	1
598.63	14.633	1	1.14	1	0.95	1	1	1	1.15	0.85	0.88	0.81	1	1	1	1	0.86	1	1.0847	1.0991	1.1609	983.39	220.13	431.09	847.07	193.08	0	0	0	1	0

The Trained Neural Network



Risk Prediction Model based on **Open Source Projects**

1.Explore possibility of building risk prediction models for open source projects.

1.Two open source projects

I. Carbon Data.

II. Fabric Java.

2.What data can be used (Jira and Github are commonly used software engineering tools. Data derived platform is available for production)

1. Jira Repos

I. Release and Milestones.

Related factors in prediction model: ***Phase***

II. Effort tracking report.

Related factors in prediction model: ***Effort*** (actual effort), ***Estimated_Effort*** (estimated effort)

III. Issue tracking reports: issue creation and resolution reports.

Related factors in prediction model: ***ISS_Num_Resolved, ISS_Num_Unresolved, Accu_Trivial, Accu_Minor, Accu_Major, Accu_Critical, Accu_Block***

IV. Personnel and Contributions.

Related factors in prediction model: ***Personnel***

V. System Modules.

2.Github

I.Commits.

Related factors in prediction model: ***CSmell, SVul***

3.How risk can be evaluated (in following slides).

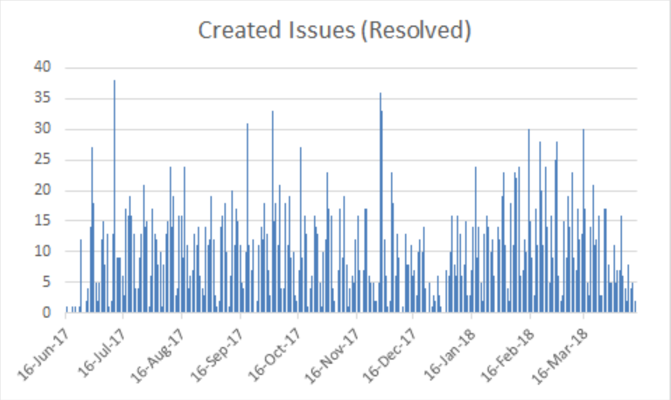
Empirically collected data - Issue Resolution (Jira)

1. The distribution of creation time indicates the phases - ***Phase***. (If actual definition of phases is available, it would be better)
2. Issue records provides creation times and resolution times of the issues.

- ***ISS_Num_Resolved, ISS_Num_Unresolved*** are calculated by ***Phase*** using the issue report

Data Table

Period	Created Issues (Unresolved)	Created Issues (Resolved)
16-Jun-17	4	1
17-Jun-17	0	0
18-Jun-17	0	0
19-Jun-17	0	1
20-Jun-17	0	1
21-Jun-17	0	0
22-Jun-17	0	1
23-Jun-17	0	12
24-Jun-17	0	0
25-Jun-17	0	0
26-Jun-17	0	2
27-Jun-17	0	4
28-Jun-17	5	14
29-Jun-17	12	27
30-Jun-17	4	18
1-Jul-17	1	5
2-Jul-17	1	2



Example of issue related datasheet - creation times

Data Table

Period	Issues Unresolved	Total Age	Avg. Age
17-Jun-17	989	109593	110
18-Jun-17	985	110017	111
19-Jun-17	976	109985	112
20-Jun-17	976	110902	113
21-Jun-17	976	111873	114
22-Jun-17	975	112763	115
23-Jun-17	970	113505	117
24-Jun-17	979	114176	116
25-Jun-17	979	115155	117
26-Jun-17	979	116133	118
27-Jun-17	978	116986	119
28-Jun-17	981	117834	120
29-Jun-17	990	117939	119
30-Jun-17	1022	118560	116
1-Jul-17	1042	119601	114
2-Jul-17	1041	120616	115
3-Jul-17	1032	120978	117
4-Jul-17	1035	122010	117
5-Jul-17	1043	123039	117
6-Jul-17	1058	124096	117
7-Jul-17	1061	125129	117

Example of issue related datasheet - Age

Empirically collected data - Time Tracking (Jira)

Time tracking records provide estimated times and actual times spent on certain tasks.

- ***Effort*** (actual effort), ***Estimated_Effort*** (estimated effort) for each ***Phase*** is calculated by the effort spent on the issues that belong to the ***Phase***.

A1 | Issue Type

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Issue Type	Key	Status	Priority	Summary	Original Time Estimated (m	Σ	Estimated Time Remaining (m	Σ	Total Time Spent (mi	Σ	Accuracy	Σ	Accuracy (%	Σ		
466	Bug	CARBONDATA-2136	Open	Major	Exception displays while loading data with BAD_RECORDS_ACTION = REDIRE			0	0	370	370						
467	Bug	CARBONDATA-2132	Open	Minor	Error while loading data with insert overwrite in partitioned table			0	0	240	240						
468	Improvemer	CARBONDATA-2129	Open	Critical	Carbon should give a remind when user use old syntax to create timeseries			0	0	340	340						
469	Bug	CARBONDATA-2124	Closed	Major	data are null in streaming ingest from file source			0	0	70	70						
470	Bug	CARBONDATA-2115	Open	Minor	In some scenario aggregate query is not fetching data from aggregate table			0	0	60	60						
471	Improvemer	CARBONDATA-2093	Open	Major	Use small file feature of global sort to minimise the number of carbondata fi			0	0	310	310						
472	Improvemer	CARBONDATA-2071	Resolved	Major	Add block size to BblockletDataMap while initialising			0	0	200	200						
473	Bug	CARBONDATA-2067	Open	Major	Streaming hand off operation throw NullPointerException			0	0	40	40						
474	Bug	CARBONDATA-2062	Open	Minor	Streaming is not setting the temp location to be used during handoff			0	0	420	420						
475	Bug	CARBONDATA-2056	Open	Major	Hadoop Configuration with access key and secret key should be passed whil			0	0	140	140						
476	Bug	CARBONDATA-2045	Resolved	Major	Query from segment set is not effective when pre-aggregate table is presen			0	0	210	210						
477	Improvemer	CARBONDATA-2033	Open	Minor	support user specified segments in major compation			0	0	300	300						
478	Bug	CARBONDATA-2026	Open	Major	Many tests are failed when carbon hive metastore is enabled			0	0	100	100						
479	Improvemer	CARBONDATA-2023	Open	Major	Optimization in data loading for skewed data			0	0	1000	1000						
480	Task	CARBONDATA-2006	Open	Minor	Project level update			0	0	40	40						
481	New Feature	CARBONDATA-1994	Open	Major	Support Java SDK API			0	0	280	1930						
482	-> Sub-task	CARBONDATA-1996	Open	Major	-> Support OutputFormat for file level carbondata												
483	-> Sub-task	CARBONDATA-2025	Open	Major	-> Refactor CarbonTablePath			0		440							
484	-> Sub-task	CARBONDATA-1997	Open	Major	-> Support FileWriter Java API for file level carbondata			0		230							
485	-> Sub-task	CARBONDATA-1995	Open	Major	-> Support InputFormat for file level carbondata			0		980							
486	Bug	CARBONDATA-1993	Open	Minor	Mismatch in Carbon properties default values and corresponding template a			0	0	690	690						
487	Bug	CARBONDATA-1992	Open	Major	Remove deprecated partitionId			0	0	220	220						
488	Improvemer	CARBONDATA-1983	Resolved	Minor	Remove unnecessary WriterVo creation			0	0	60	60						
489	Bug	CARBONDATA-1971	Open	Major	Measure Null Value Recognise in blocklet prunning.			0	0	1040	1040						

Example of time tracking datasheet.

Empirically collected data - Code Analysis (Github)

Code metrics applied to each commit to measure code quality and technical debt.

- Number of Code smells (***CSmell***) is calculated by commits that belong to ***Phase***
- Number of Vulnerabilities (***SVul***) is calculated
- by commits that belong to ***Phase***

* This analytical data is generated by SQUAAD

Code Metrics

	A	B	C	D	E	F	G	
1	application	csha	cwhen	message	branch	vulnerabilities	code_smells	f
2	apache-carbondata	ceac8abf6	4/9/2018 4:40	[CARBON	refs/head:	171	2490	
3	apache-carbondata	e26cccc41	4/2/2018 5:38	[CARBON	refs/head:	171	2489	
4	apache-carbondata	cf1e4d4ca	4/6/2018 1:40	Blocklets	refs/head:	168	2495	
5	apache-carbondata	ecd6c0c5·	4/15/2018 19:55	[CARBON	refs/head:	168	2493	
6	apache-carbondata	4c9bed8b	4/3/2018 3:48	[CARBON	refs/head:	167	2491	
7	apache-carbondata	9ee74fe07	4/13/2018 0:14	[CARBON	refs/head:	167	2491	
8	apache-carbondata	52048183	4/8/2018 4:44	[CARBON	refs/head:	167	2491	
9	apache-carbondata	cfb8ed9f5	4/1/2018 1:30	[CARBON	refs/head:	167	2491	
10	apache-carbondata	13cdeb9f4	4/1/2018 5:08	[CARBON	refs/head:	167	2491	
11	apache-carbondata	359f6e6b2	4/10/2018 7:12	[CARBON	refs/head:	167	2481	
12	apache-carbondata	df8f06739	4/8/2018 3:05	[CARBON	refs/head:	167	2482	
13	apache-carbondata	b439b00f(	3/13/2018 23:31	[CARBON	refs/head:	167	2481	
14	apache-carbondata	f6990d62:	4/7/2018 20:01	[CARBON	refs/head:	167	2481	
15	apache-carbondata	638ed1fa7	3/29/2018 4:50	[CARBON	refs/head:	167	2481	
16	apache-carbondata	b52f1571(	3/25/2018 1:15	[CARBON	refs/head:	165	2477	
17	apache-carbondata	280a4003	4/5/2018 2:54	[CARBON	refs/head:	165	2475	
18	apache-carbondata	55084602	3/22/2018 23:42	[CARBON	refs/head:	164	2460	
19	apache-carbondata	fb1516c0(	3/18/2018 19:23	[CARBON	refs/head:	164	2460	
20	apache-carbondata	6374d361	3/31/2018 7:42	[CARBON	refs/head:	164	2459	
21	apache-carbondata	0992b3b2	4/1/2018 10:09	[CARBON	refs/head:	164	2459	
22	apache-carbondata	f910cfa98	3/24/2018 7:38	[CARBON	refs/head:	164	2459	
23	apache-carbondata	cd509d5d	3/30/2018 19:42	[CARBON	refs/head:	164	2461	
24	apache-carbondata	e8da8800	2/5/2018 1:10	[CARBON	refs/head:	164	2461	
25	apache-carbondata	9fba6845·	3/26/2018 3:17	[CARBON	refs/head:	164	2461	
26	apache-carbondata	5daae951	3/28/2018 4:08	[CARBON	refs/head:	163	2451	
27	apache-carbondata	7e0803fec	3/22/2018 21:13	[CARBON	refs/head:	163	2451	
28	apache-carbondata	0c200d83	3/26/2018 4:06	[CARBON	refs/head:	164	2444	
29	apache-carbondata	877eabdd	3/26/2018 23:50	[CARBON	refs/head:	164	2444	

Example of github commits datasheet.

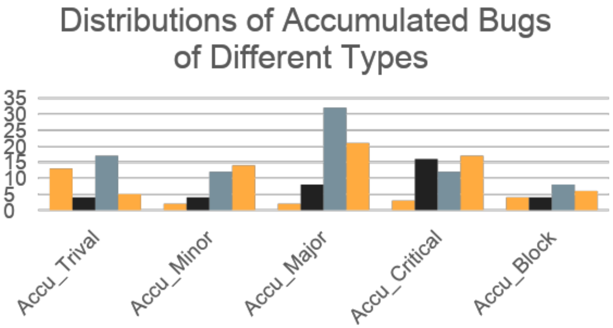
Classified Cumulative Bugs by Severity (Jira)

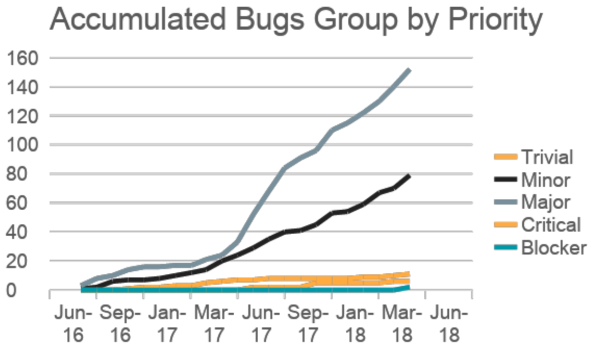
Accumulated Bugs of different severity reflect the potential technical debt.

- ***Accu_Trivial, Accu_Minor, Accu_Major, Accu_Critical, Accu_Block*** are determined by categorizing the bugs according to their severity.

D	E	F	G	H
Accu_Trival	Accu_Minor	Accu_Major	Accu_Critical	Accu_Block
13	2	2	3	4
4	4	8	16	4
17	12	32	12	8
5	14	21	17	6

Example of empirically collected data for accumulated bugs of different types



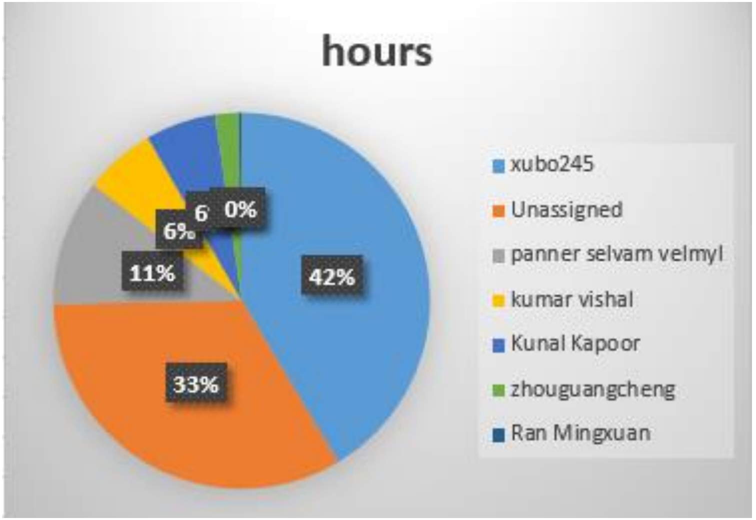


Example of distributions of categorized bugs

Empirically collected data - Personnel & Contribution (Jira)

Effort distribution over team members identify the contributors, which helps measure the balance of workload.

- The distribution of the effort to personnel determines the core contributors, the number of which define the factor ***Personnel***. Contributors who contribute larger than 5% are determined as core contributors.



Effort Distribution to Different Developers

	A	B	C
1	name	hours	%
2	xubo245	3438	0.41
3	Unassigne	2762	0.33
4	panner se	904	0.1
5	kumar visl	504	0.06
6	Kunal Kap	502	0.06
7	zhouguan	168	0.02
8	Ran Ming›	20	0
9	tianli	2	0
10	Zuo Wang	0	0
11	Zhichao Zl	0	0
12	zhaowei	0	0
13	zhangwei	0	0
14	zhangshuı	0	0
15	Yadong Qi	0	0
16	xuchuanyi	0	0
17	xbkaishui	0	0
18	WilliamZh	0	0
19	Weizhong	0	0
20	wangsen	0	0
21	Vinod Roh	0	0
22	Vinod KC	0	0

Example of identifying core developers

Risk Assessment Metrics based on Open Source Data

The proposed method to measure risk.

1. ***Risk_inflation_rate*** = ***ISS_Num_Unresolved*** (number of unsolved issues) / ***ISS_Num_Resolved*** (number of solved issues) (at the end of a milestone)
2. ***Pressure*** = ***Actual_Effort*** / ***Estimated_Effort*** (for each phase)
3. ***Risk*** = ***Risk_inflation_rate*** + ***Pressure***

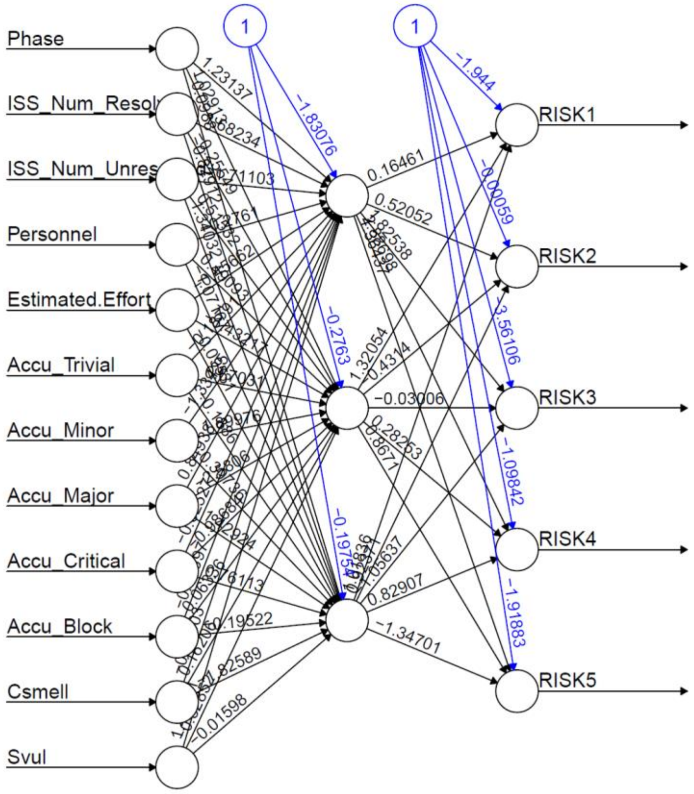
Phase	ISS_Num_Resolved	ISS_Num_Unresolved	Accu_Trival	Accu_Minor	Accu_Major	Accu_Critical	Accu_Block	risk_inflation_rate	Effort	Difference	Personnel	Estimated Effort	Pressure	Risk
July/16/2017	231	109	13	2	2	3	4	0.471861	4876	4876	16	3840	1.269792	1.741653
Sep/16/2017	441	148	4	4	8	16	4	0.335601	11230	6354	16	7680	0.827344	1.162945
Feb/16/2018	1341	503	17	12	32	12	8	0.375093	33871	22641	16	19200	1.179219	1.554312
Mar/16/2018	441	571	5	14	21	17	6	1.294785	46275	12404	16	3840	3.230208	4.524993

Training Risk Prediction Model

Input Example for the neural network based risk prediction model

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Phase	ISS_Num_	ISS_Num_	Personnel	Estimated	Accu_Triva	Accu_Mino	Accu_Majo	Accu_Criti	Accu_Bloc	RISK1	RISK2	RISK3	RISK4	RISK5
2	1	131	169	8	3840	0	3	0	0	0	0	0	0	1	0
3	2	281	568	8	7680	0	2	8	0	0	0	0	0	1	0
4	3	541	453	8	19200	3	12	17	0	0	0	1	0	0	0
5	4	281	271	8	13840	5	14	21	0	0	0	1	0	0	0
6	1	231	109	16	3840	13	2	2	3	4	0	1	0	0	0
7	2	441	148	16	7680	4	4	8	16	4	1	0	0	0	0
8	3	1341	503	16	19200	17	12	32	12	8	0	1	0	0	0
9	4	441	571	16	3840	5	14	21	17	6	0	0	0	0	1
10															

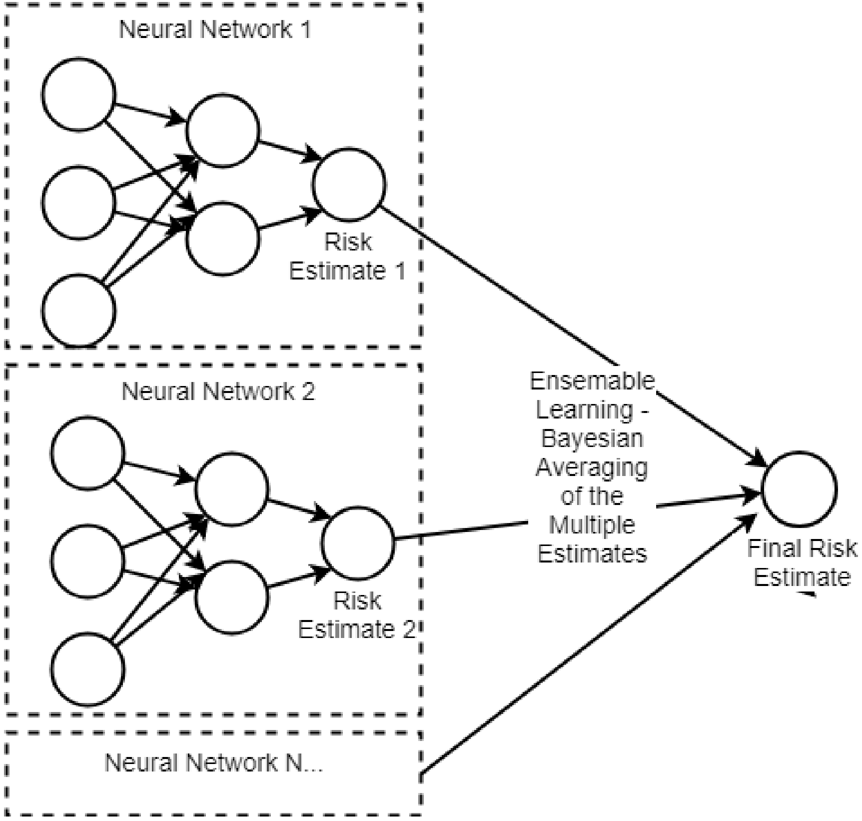
The Trained Neural Network



Combination of Risk Prediction Models

Bayesian Averaging on multiple risk estimates as the final estimate of risk

The final estimate of level of risk is the weighted average of the two modes. The weights are determined by the variances of each mode's estimates.



Model Evaluation

Bootstrapping is applied to estimate classification rate and the confidence.

- 1000 resampling from the original datasets.
- For each resample, classification rate (precision) is calculated and confidence level is estimated.
- We have reached 48% classification accuracy on average for the combined model.
- 26% - 70% classification accuracy at 95% confidence level.

	USC Model	Open Source Model	Combined Model
Classification Rate	97%	50%	48%
Std. Error.	0.017	0.177	0.112

Risk Prediction Model Prototype

Example - Risk Prediction API

Instruction: Please submit a csv file with project data specified in this file

Project Data: Choose File No file chosen

Submit


```
{
   "results": [
      {
         "risk_lvl1": "1.8995183039438e-185",
         "risk_lvl2": "5.78383891570669e-06",
         "risk_lvl3": "0.999999999590604",
         "risk_lvl4": "1.86143195634384e-128",
         "risk_lvl5": "0",
         "predicted": "3"
      }
   ],
   "report": "[1] \"prediction calculation with:\"\n  RELY DATA CPLX RUSE
PVOL ACAP APEX\n1    1 1.14    1 0.95 0.87 0.85    1\n[1] \"prediction
results:\"\n              [,1]         [,2] [,3]         [,4] [,5]\n[1,]
1.899518e-185 5.783839e-06    1 1.861432e-128    0\n     [,1]\n[1,]
3\n"
}
```

Intelligent Risk Mitigation

How to mitigate risk after the risk is predicted?

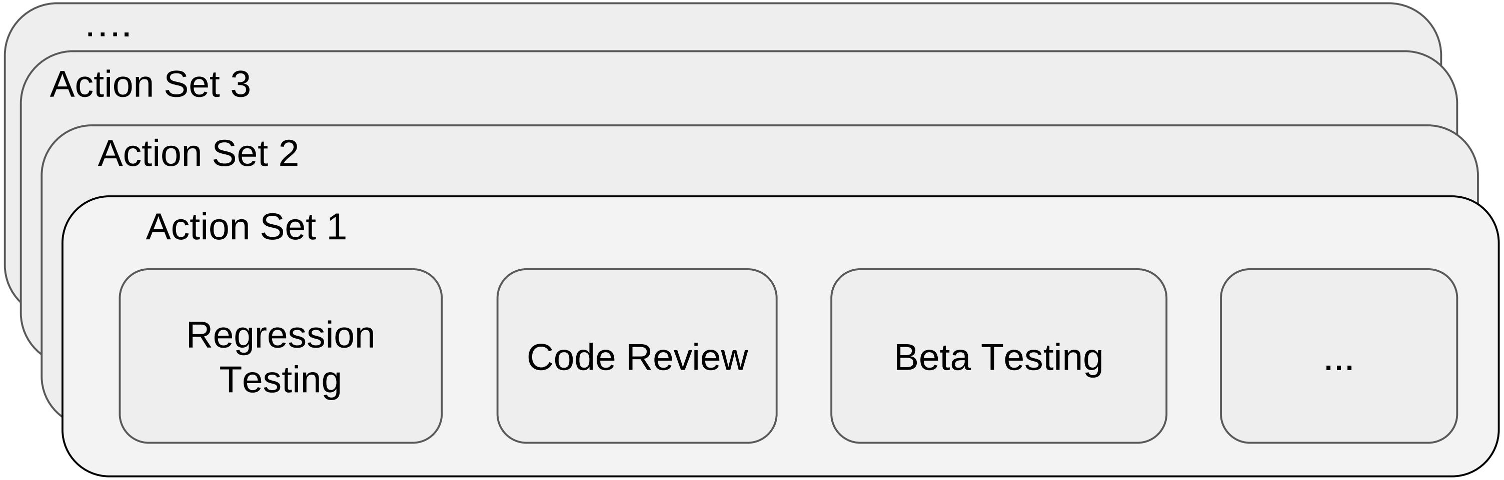
1. If I’m not experienced.
2. If I just want reduce risk to certain level.
    a. I want to take the risk for certain purposes.

Knowledge Base for Mitigating Risk

1. Common Practice for Mitigating Risk of Different Kinds
   a. The practices suggested by COQUALMO: Automated analysis, peer reviews, execution testing tools may be effective for different kinds of defects.
2. Learning from Daily Practice
   a. E.g certain form of code review conducted by the organization may be effective for architectural technical debt.

Risk Mitigation Strategies

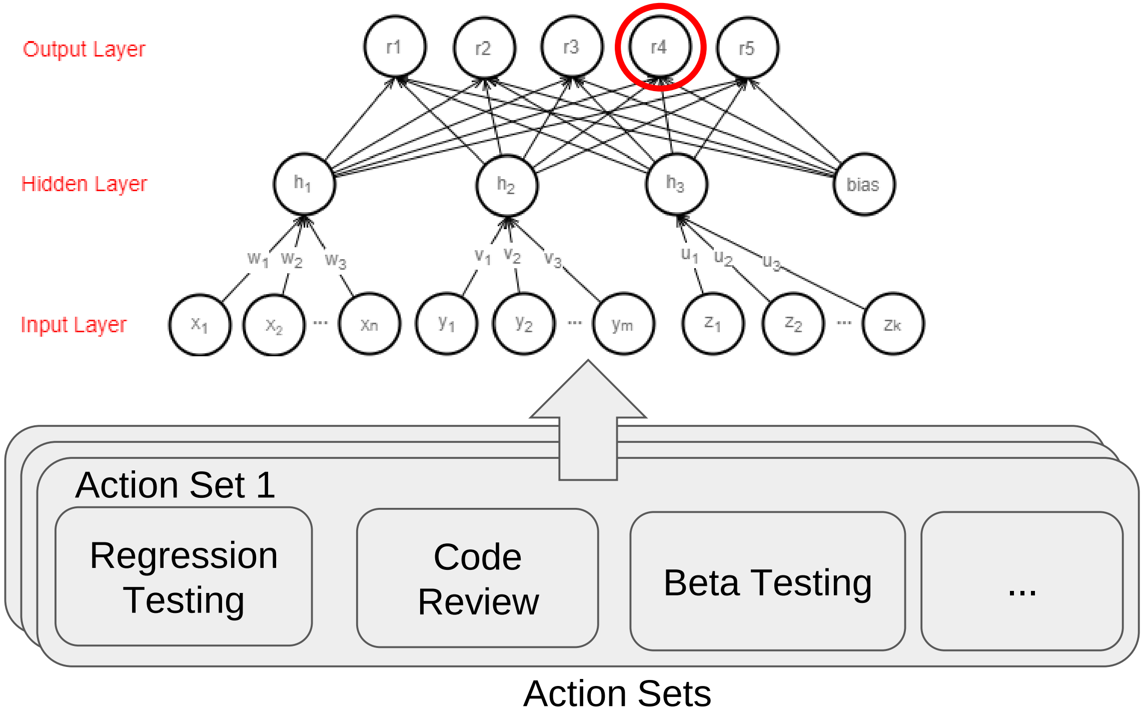
A set of actions to mitigate risk at certain phase of a project is a risk mitigation strategy. At certain point of a project, there may be multiple risk mitigation action sets that can be applied to reduce the predicted risk level.



Strategies

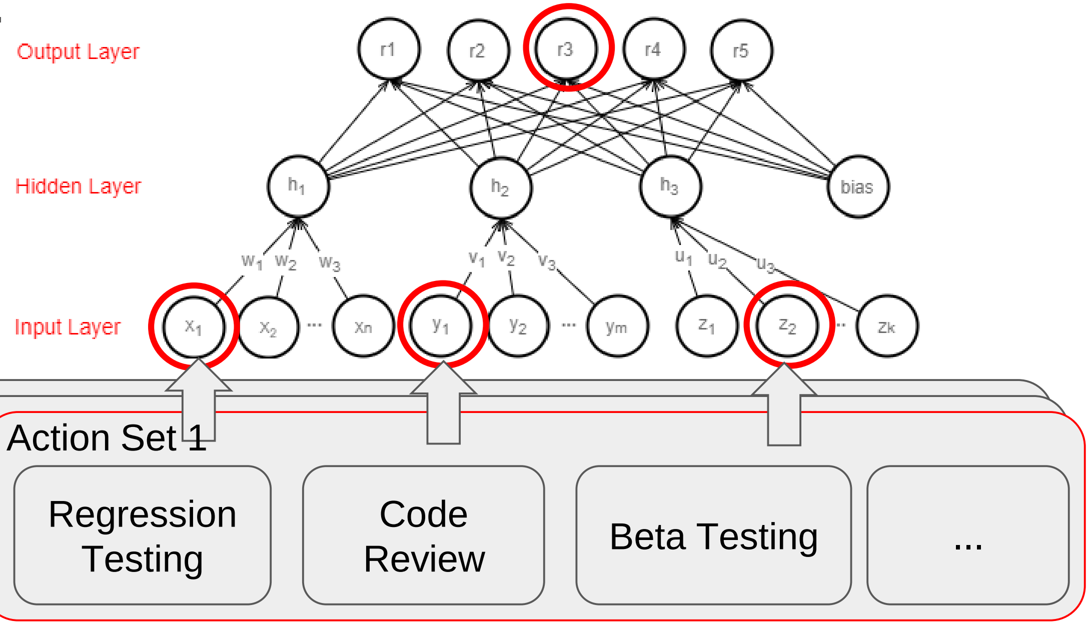
Learning the influences on risk

The effects of the candidate actions on risk are modeled as influences on factors of risk prediction model.



Select the best actions to mitigate risk

Select the best action set to mitigate risk to a certain risk level by the risk prediction model.



Action Sets

Demo

Conclusions

Operational Concept

- Developed the UI prototype and risk prediction model to better explain the results/target;

Feasibility Evidence

- Defined and trained the risk prediction model with available data;
- Proved the technical feasibility for model training, optimization and integration to the current system.

Next Steps for Future Phases

1. Optimize the risk prediction model based on real data from pilot projects
    - Some management dataset may be available from certain conference, for example, ASE.

2. **Define the risk mitigation actions based on development business units input.**

3. Consider the possibility of integrating natural language processing of unstructured data from reports/reviews/feedback - those data may be related to risk.

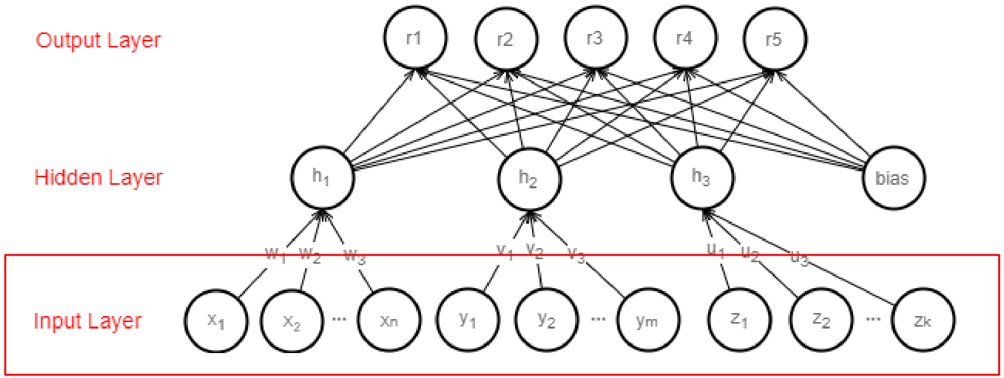
References

1. Improving the accuracy of COCOMO's effort estimation based on neural networks and fuzzy logic model.
2. Neural Network Model For Software Size Estimation Using Use Case Point Approach
3. Model selection for Neural Network Classification
4. https://en.wikipedia.org/wiki/Artificial_neural_network
5. Feature selection with neural networks
6. Confidence Estimation Methods for Neural Networks

Backup Slides

The Three Layers - Input Layer

- **Group X:** results from SQAaaS(Code Repository Static Analysis)
  - Code Smell (CSmell) and Security vulnerability (SVul)
- **Group Y:** project settings from COCOMO and COQUALMO
  - 22 project factors and 3 defect removal factors are used.
- **Group Z:** runtime states from Quality Management Database and Evidence Documents
  - FCR (feature completion rate), CD (number of code defects), DRR (defect removal rate), ISS (number of reported issues at the feature level), ISRR (issue removal rate).



The Three Layers - Hidden Layer

- Neurons in the hidden layer are the generalised representations of the inputs to reflect their influences on risk from different aspects. They are the linear combination of neurons in the input layer.
  - h1 models the influence from technical debt.

$$h_1 = w^T * x$$

  - h2 models the influence from the general project characteristics, for example, ability to solve risk.

$$h_2 = v^T * y$$

  - h3 models the current status of the project, for example, the progress of the project (avoid being over schedule or budget).

$$h_3 = u^T * z$$

The Three Layers - Output Layer

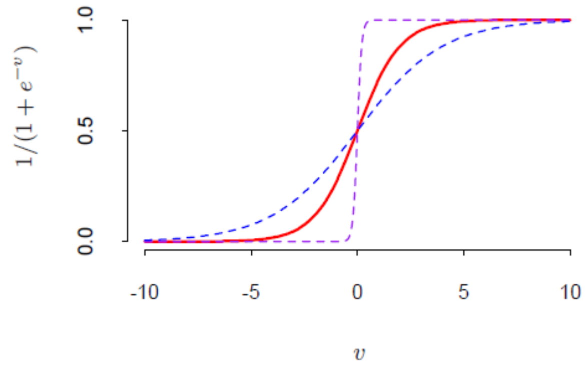
- Five units in the output layer model the five levels of risk.
- The five levels of risk correspond to the five levels of risk exposure outlined in the Risk Calculator.
- The target t_k are the linear combination of the inputs from the hidden layer nodes plus the bias.

$$T_k = bias + \beta_k^T h$$

- The probability of each level of risk is calculated by logistic activation function. The most probable risk is output as the estimated risk level.

$$g_k(x) = \frac{1}{1 + e^{-T_k}}$$

$$G(x) = argmax g_k(x)$$

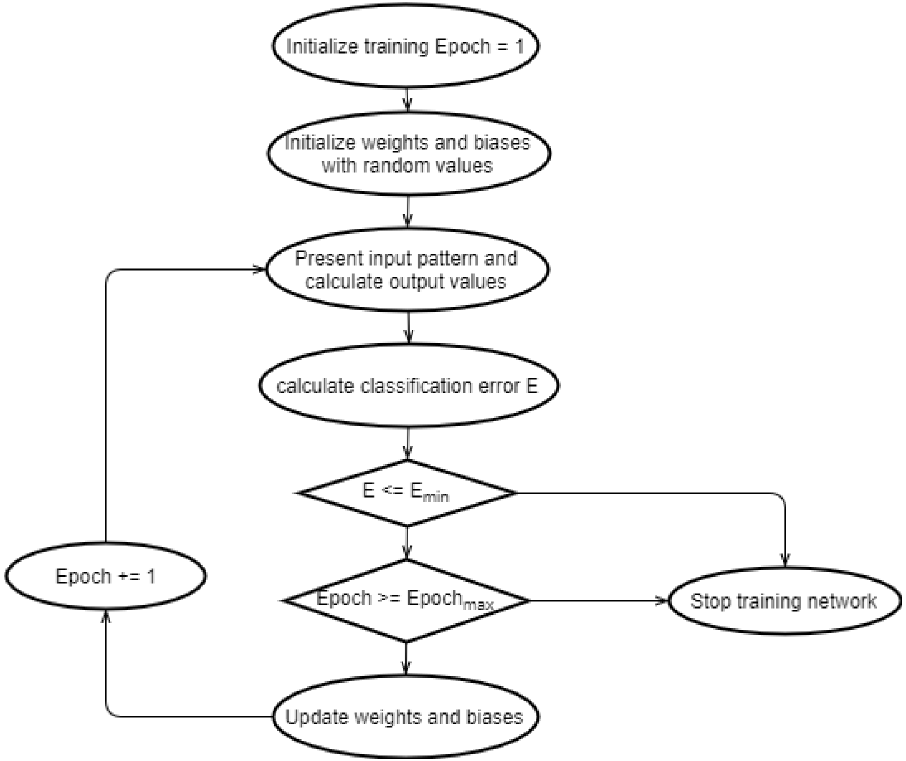


Neural Network Training and Testing Technical Details

Example of the Input Data

	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF
1	SVul	RELY	DATA	CPLX	RUSE	DOCU	TIME	STOR	PVOL	ACAP	PCAP	PCON	APEX	PLEX	LTEX	TOOL	SITE	SCED	AA	PR	ETT	FCR	CD	ISS	ISRR	DRR	RISK1	RISK2	RISK3	RISK4	RISK5
2																															
3																															
4																															
5																															
6																															
7																															
8																															
9																															
10																															
11																															
12																															
13																															
14																															
15																															
16																															
17																															
18																															
19																															

The Model Training Process



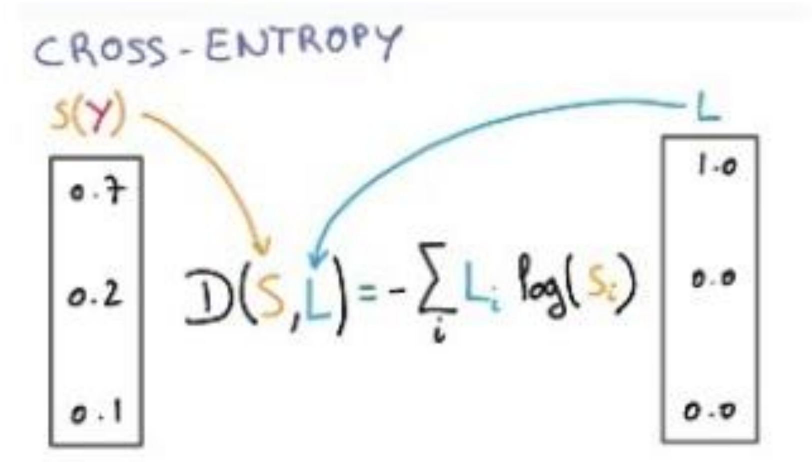
Model Training Method

- Cost Function
  - To train the classifier, cross entropy cost function is used to model the error (the difference the actual and the predicted value).
- Backpropagation
  - As the sample data input into the model, the error is propagated backwards to change the weights of the specific neurons, specifically is to minus a learning rate multiplied by the partial derivative of the error with respect to the weight.
  - The prediction will converge to the actual as more data points is used to train the model.

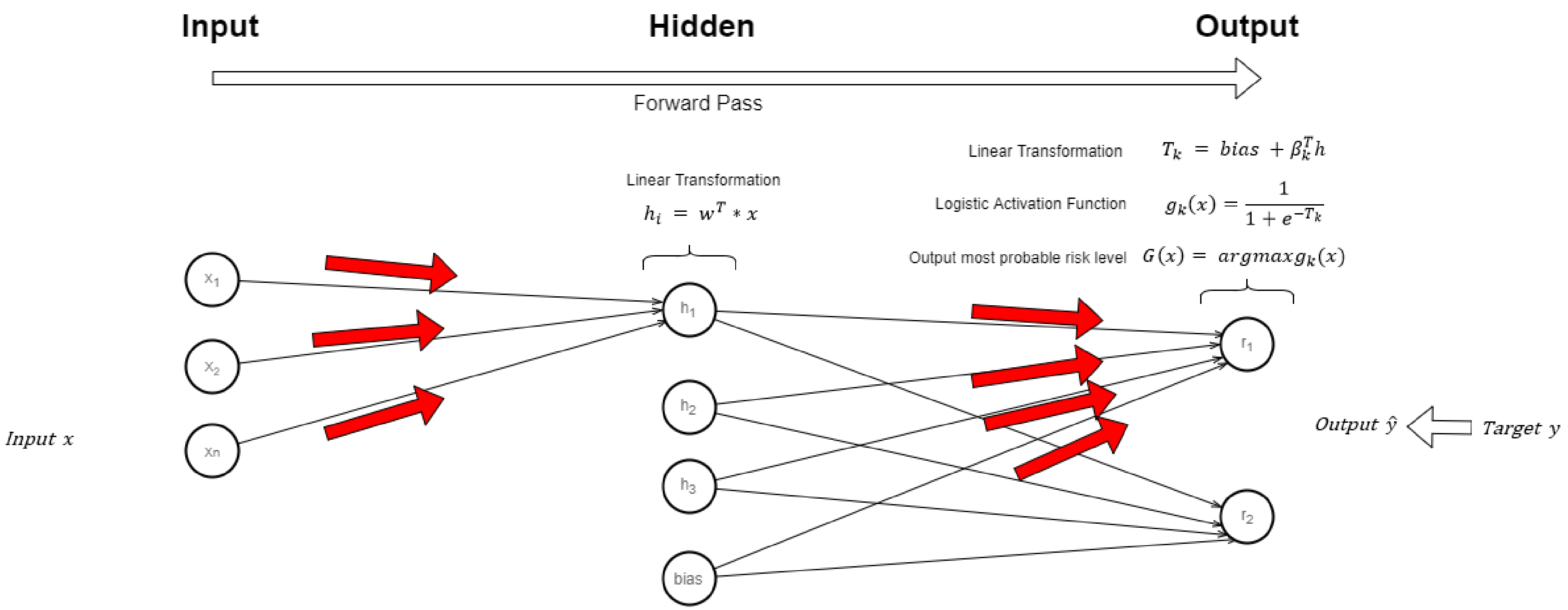
Cross Entropy as the Cost Function

- Distance Measure between f(x_i) and Labels

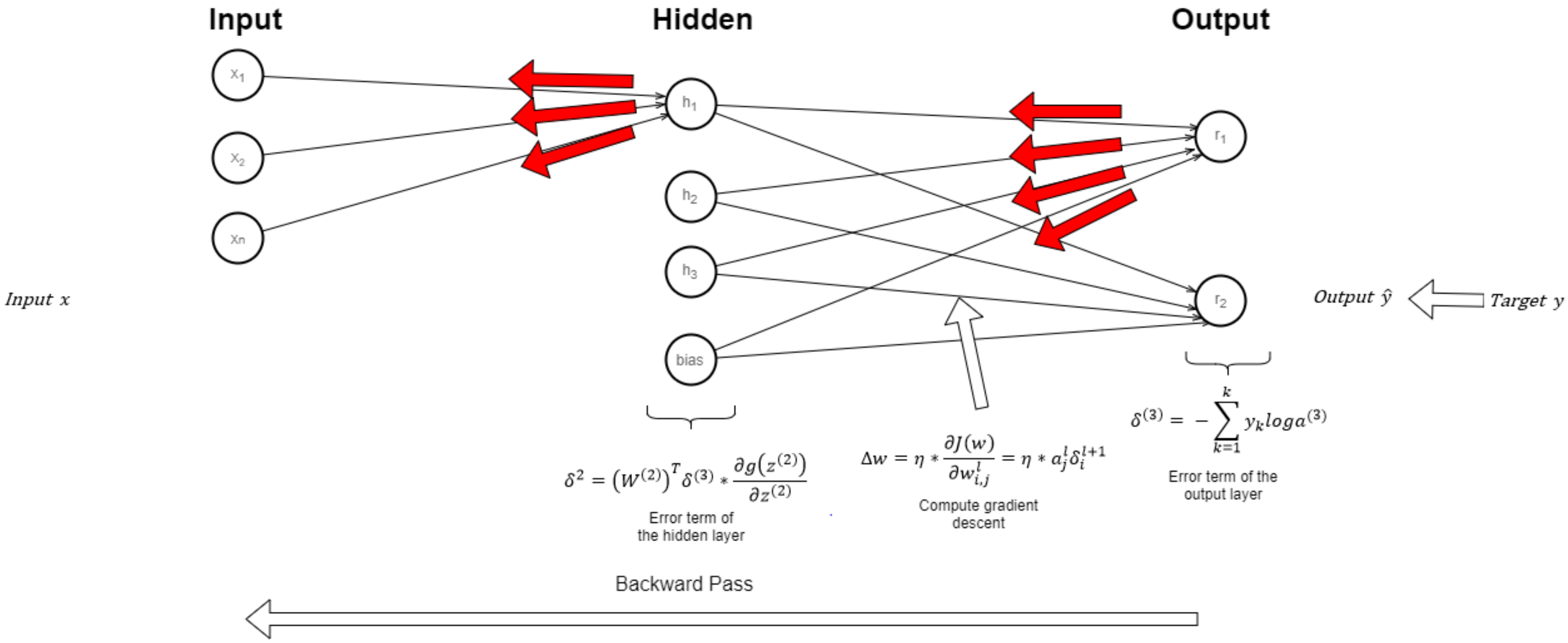
$$R(\theta) = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik} log f_k(x_i)$$



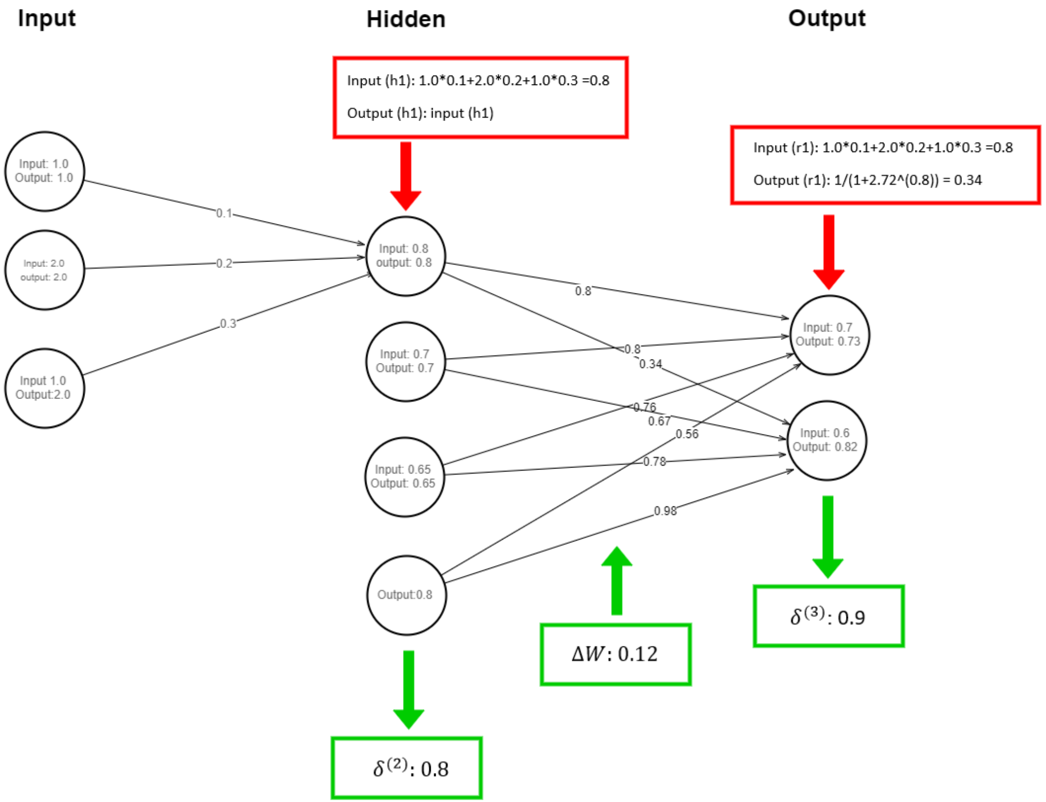
Forward Propagation



Backward Propagation



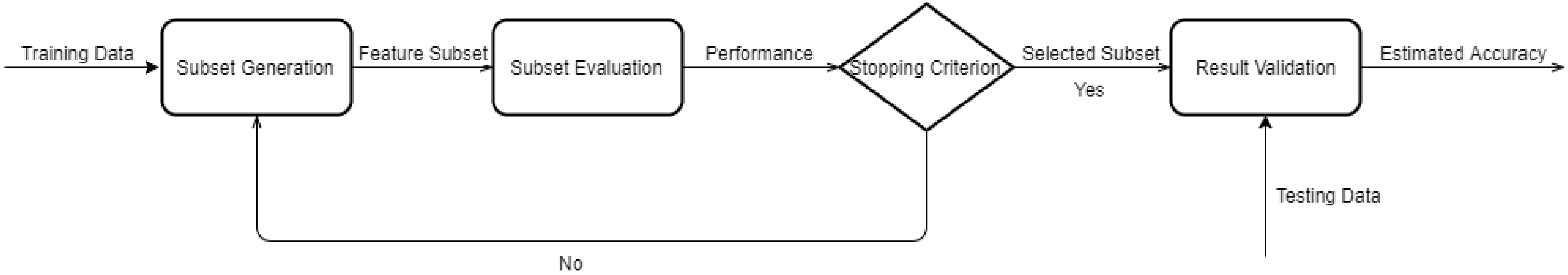
An Example of Backpropagation



Model Validation

- A portion of the data should be used as the validation data set to decide the hyper-parameters of the neuron network, for example, the number of neurons.
- If adding or removing a factor doesn’t improve the performance of the model significantly, we may remove the factor to avoid overfitting. The potential criteria are E (Classification Error Rate), BIC (Bayesian Information Criteria), AIC (AKaike Information criteria), etc.

Feature Selection

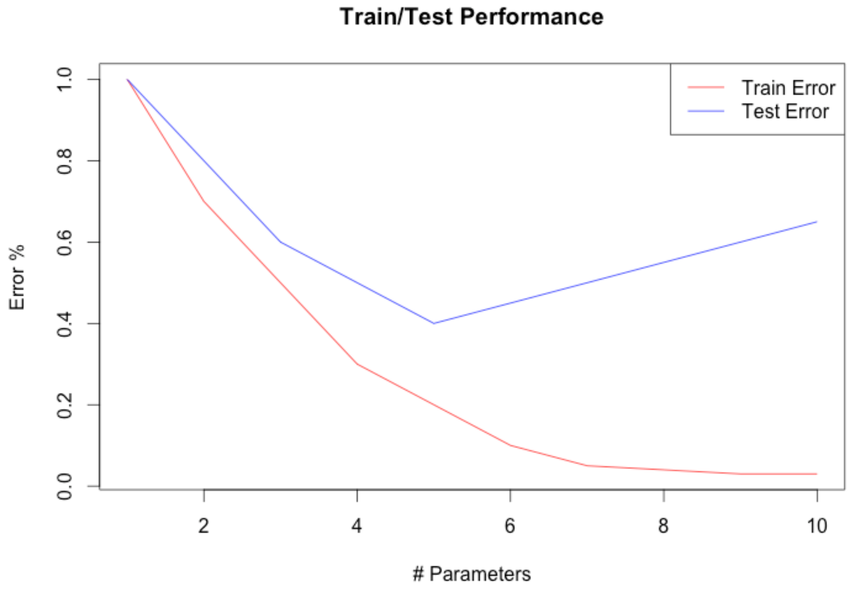


Feature Selection

1. Iteratively drop the features of the trained neural network, and assess the classification accuracy against the validation data set.
2. Calculated the expected classification accuracy.
3. Rank the features with their classification accuracy.
4. Eliminate the least salient features according to the rank.
5. Calculate the drop in classification accuracy \delta A. If \delta A is smaller than \delta A_0, then output the most parsimonious set of features, otherwise go back to step 4.

Error Rate during Feature Selection

- To avoid the issue of overfitting, the number of variables in the final model equals the minimum number of parameters that produces minimum error minus 1.



Model Testing

- For this classification model, classification error is used to evaluate the estimation accuracy.
- An independent set of data need to be used to test the estimation accuracy of the model.
- Cross validation or bootstrap may also be used if no enough data points available for accuracy estimation.
- Using bootstrap to resample from the data set to derive sampling distribution of accuracy criterion to estimate the confidence interval.

Estimation of Accuracy Confidence by Bootstrapping

- After 1000 (or other large number) runs of resampling from the data set, the distribution of classification accuracy can be approximated by sampling distribution.
- The confidence interval of classification error rate can be estimated.