Design patterns in insurance cost prediction

Using Python, Django and React

Christopher Mason

Counsyl

Outline

- About Counsyl
- How/why estimate insurance cost?
- Key Challenges
- Estimate Process and Models
- Key Patterns and anti-Patterns
- Lessons learned

About Counsyl

- We offer DNA screening for men, women, and their children.
- Our focus is on diseases where advanced knowledge makes a difference in health outcomes.
- We strive to make genetics routine in clinical practice.

Counsyl Products

Foresight™ Carrier Screen

Learn about over 175 inherited diseases you and your partner might carry that could affect your future family.

Prelude™ Prenatal Screen

Non-invasively test for chromosomal conditions early in your baby's development.

Reliant™ Cancer Screen

Learn about your risk of developing certain preventable cancers before they happen so you can be prepared.

Our tech stack

python™

django

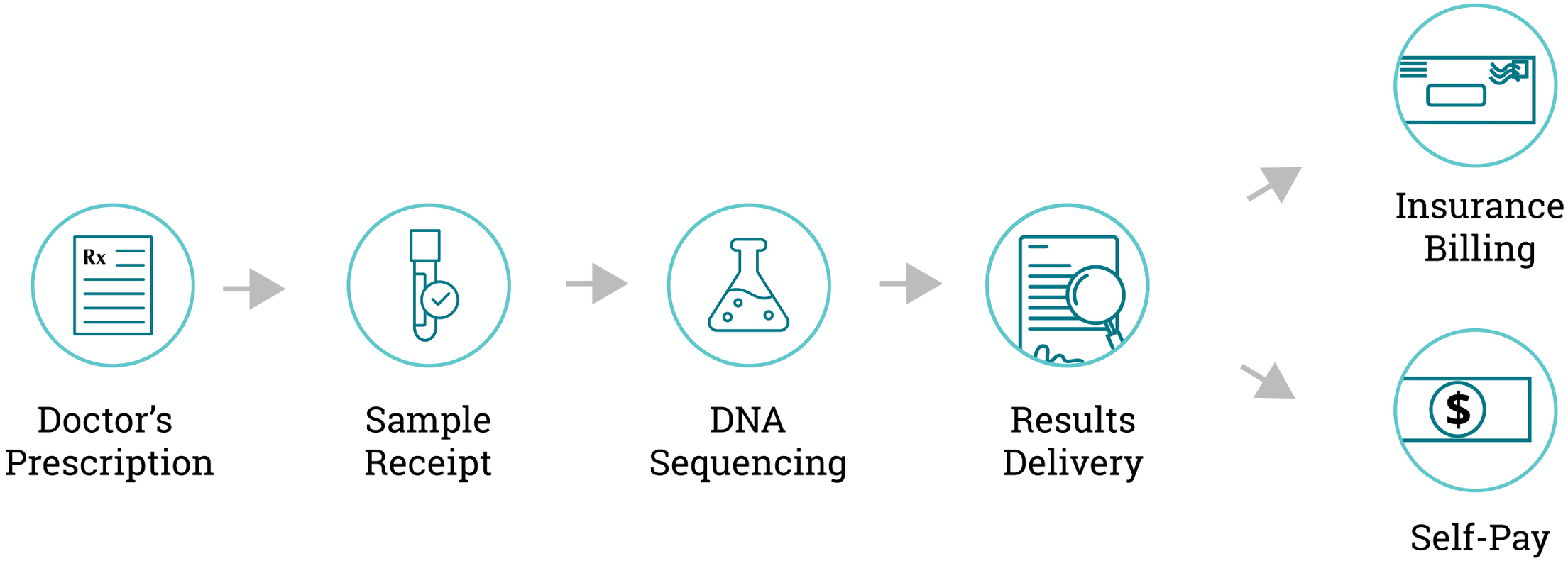
Celery

PostgreSQL

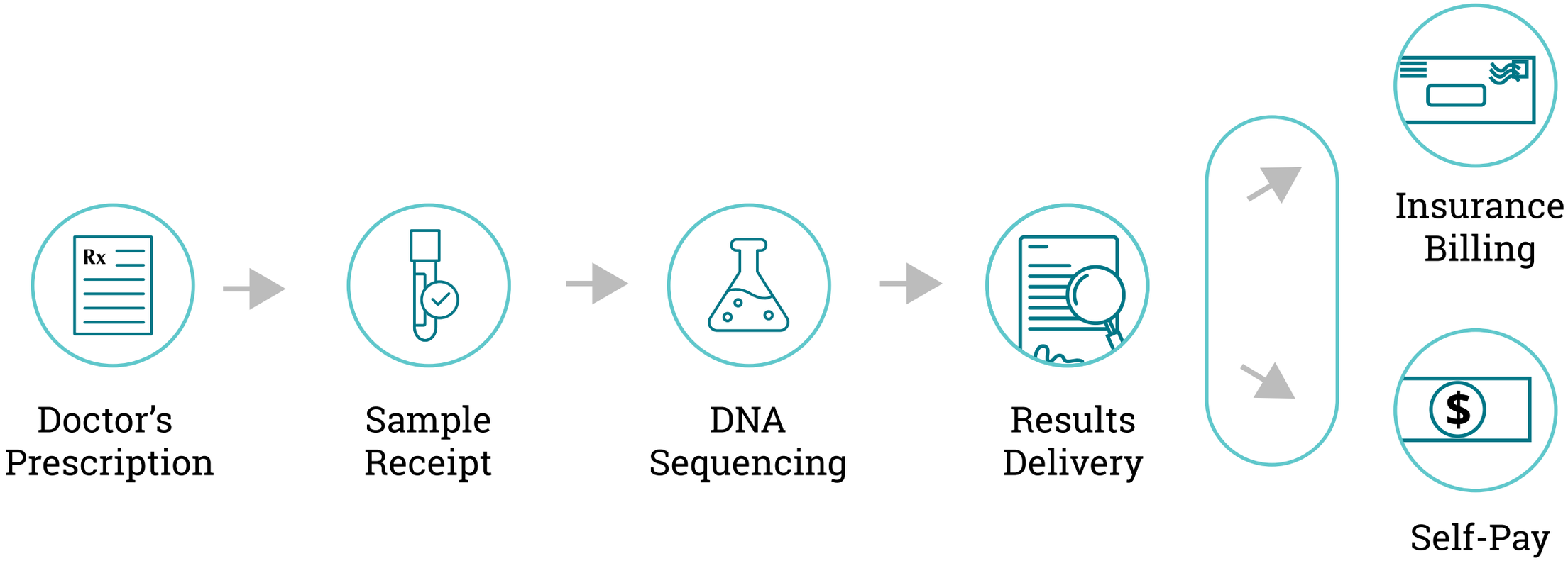
redis

React

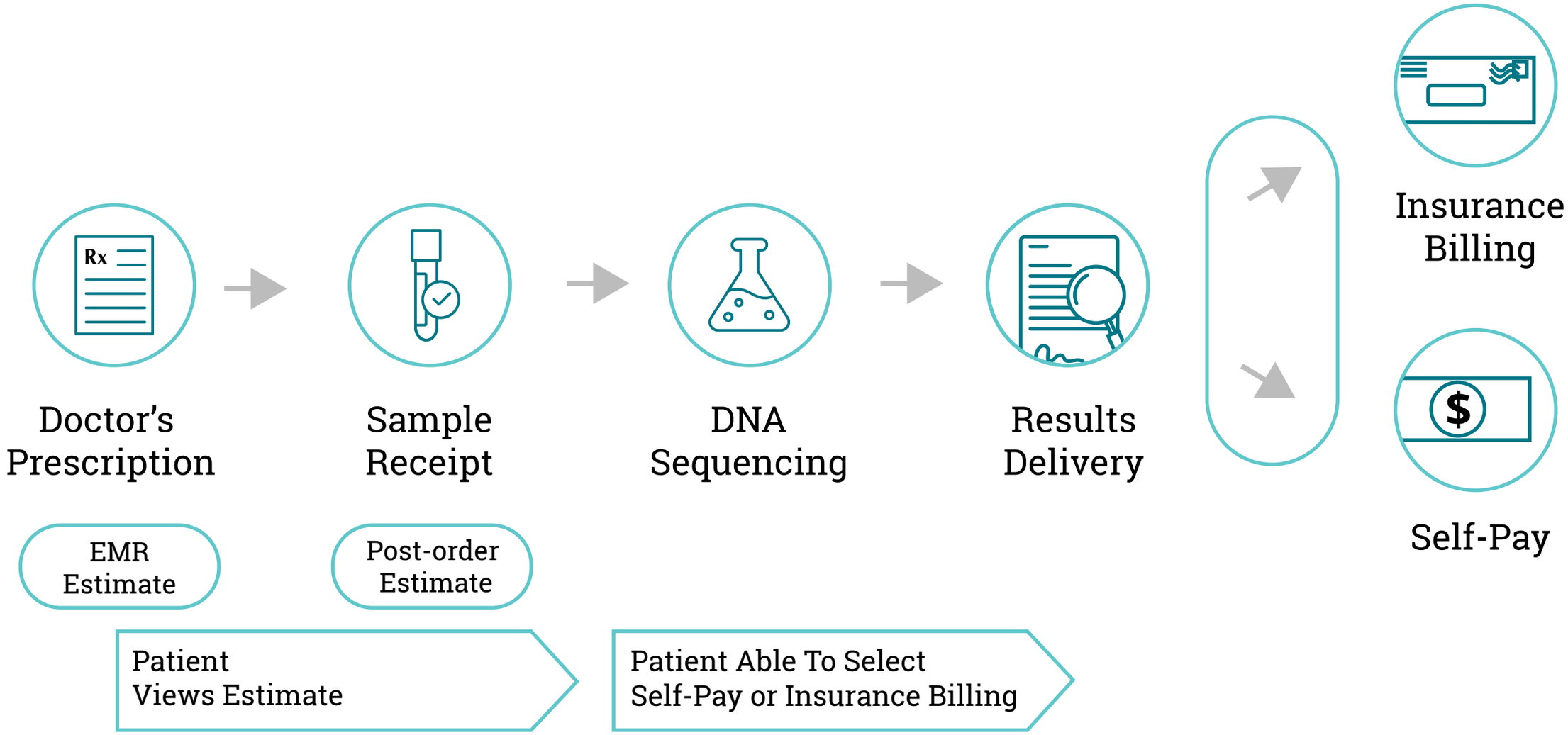
How Counsyl works



How Counsyl works



How Counsyl works



How Insurance Works

Test: Gene Panel: 175+ Diseases

How Insurance Works

Test: Gene Panel: 175+ Diseases

Claim: Counsyl → Insurance Company

Disease	CPT Code	Charged
Cystic Fibrosis	81220	$299.00
Gaucher disease	81251	$ 47.25
...	...	...

How Insurance Works

Test: Gene Panel: 175+ Diseases

Claim: Counsyl → Insurance Company

Disease	CPT Code	Charged
Cystic Fibrosis	81220	$299.00
Gaucher disease	81251	$ 47.25
...	...	...

Insurance Benefits: @ Insurance Company

Deductible $100.00

Co insurance 10%

How Insurance Works

Test: Gene Panel: 175+ Diseases

Claim: Counsyl → Insurance Company

Disease	CPT Code	Charged
Cystic Fibrosis	81220	$299.00
Gaucher disease	81251	$ 47.25
...	...	...

Insurance Benefits: @ Insurance Company

Deductible $100.00
Co insurance 10%

Explanation of Benefits (EOB): Counsyl ← Insurance Company

CPT Code	Charged	Allowed	Paid	Deductible	Coinsurance
81220	$299	$254	$138	$100	$15
81251	$ 47	$ 0			
...	...	...	...		

How Insurance Works

Test: Gene Panel: 175+ Diseases

Claim: Counsyl → Insurance Company

Disease	CPT Code	Charged
Cystic Fibrosis	81220	$299.00
Gaucher disease	81251	$ 47.25
...	...	...

Insurance Benefits: @ Insurance Company

Deductible $100.00
Co insurance 10%

Explanation of Benefits (EOB): Counsyl ← Insurance Company

CPT Code	Charged	Allowed	Paid	Deductible	Coinsurance
81220	$299	$254	$138	$100	$15
81251	$ 47	$ 0			
...	...	...	...		

Appeals / Followup Counsyl → Insurance Company

Check Counsyl ← Insurance Company

What are we estimating?

Patient Responsibility = **Deductible**

What patient owes

Fixed amount patient owes out-of-pocket each year before insurer will pay for any health expenses

+ **Coinsurance**

After deductible is met, patient owes a certain percentage of all medical charges

+ **Copay**

For every medical service rendered, patient owes a fixed, usually small, dollar amount

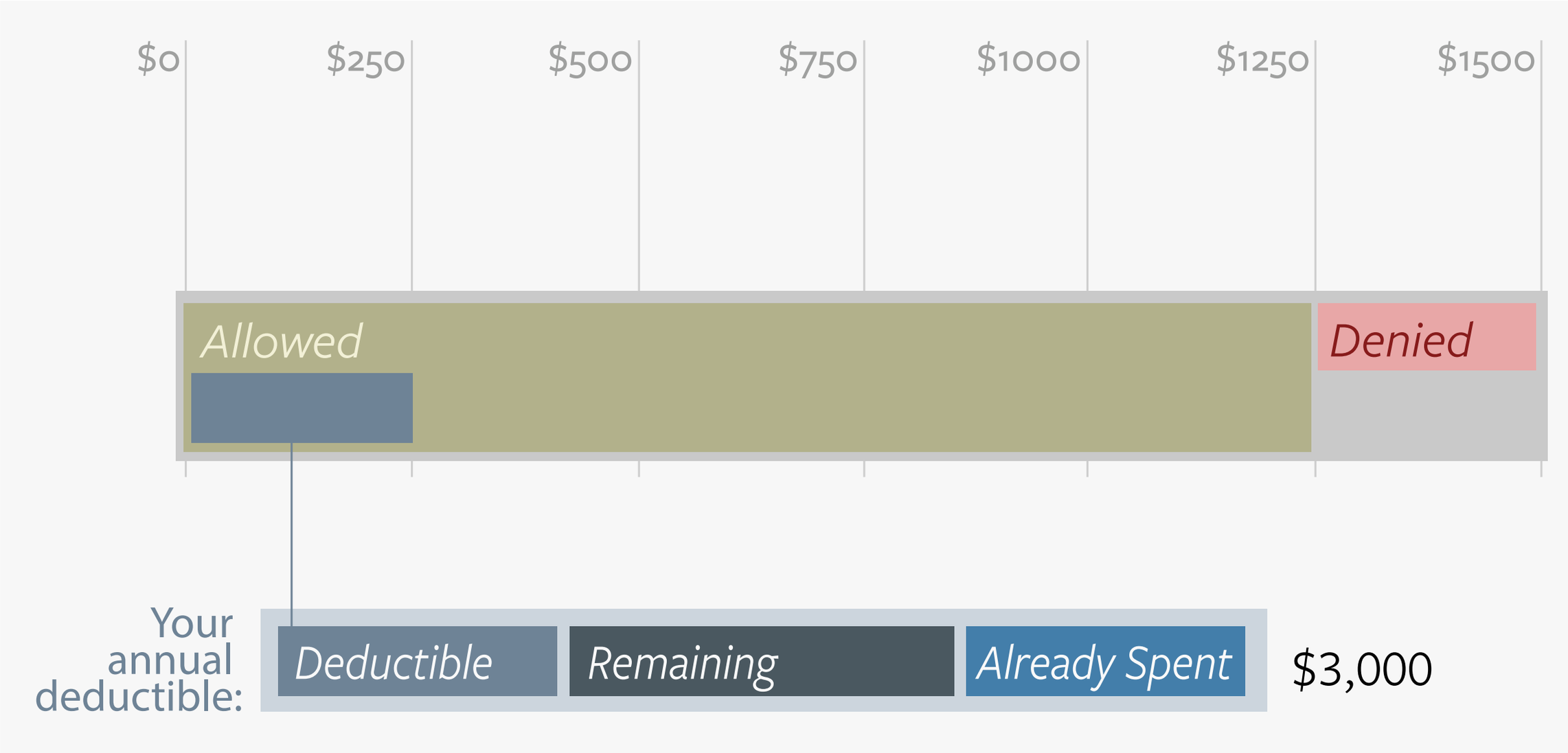
Allowed = Amount insurance thinks our test is worth

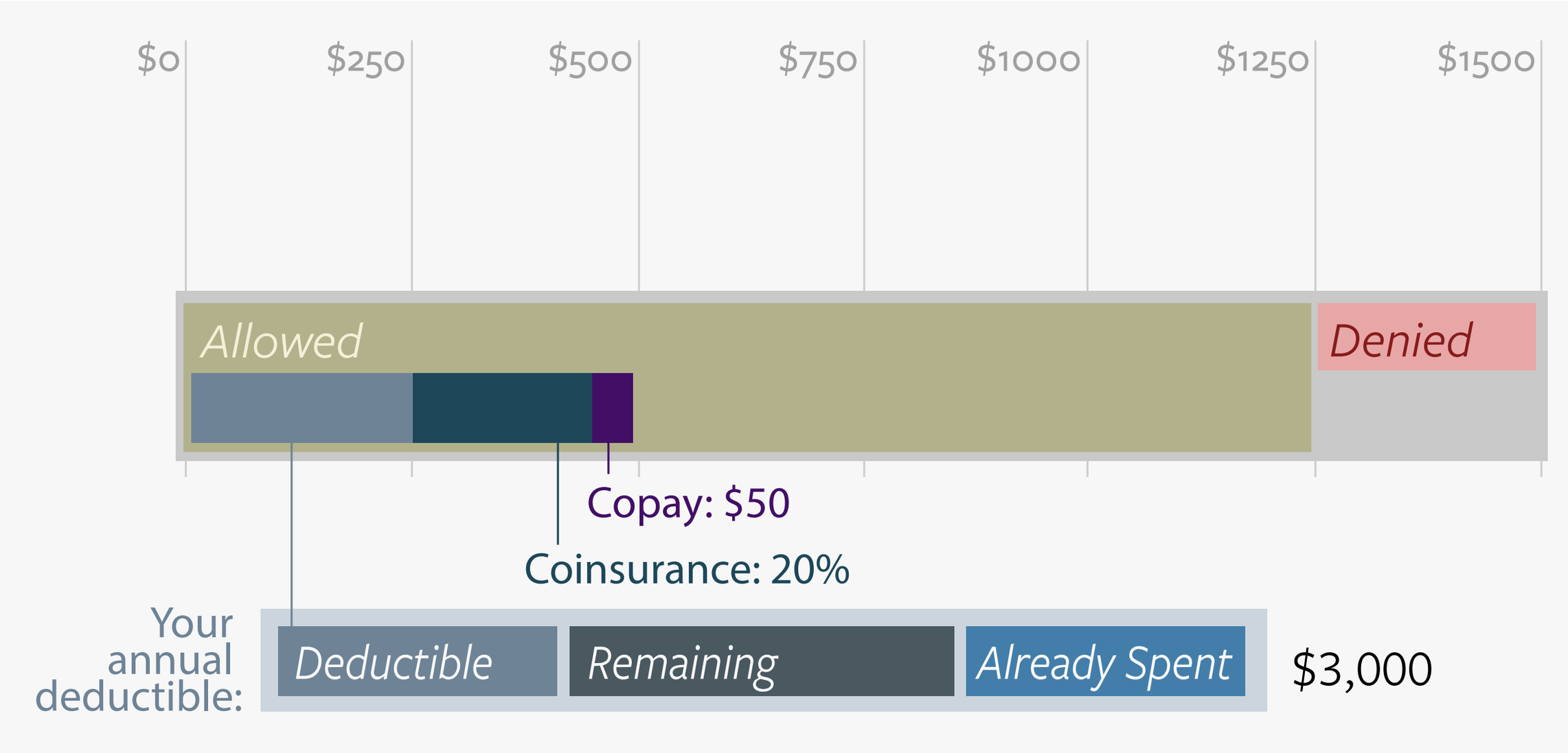
Total Charges = Allowed (PR+Paid)

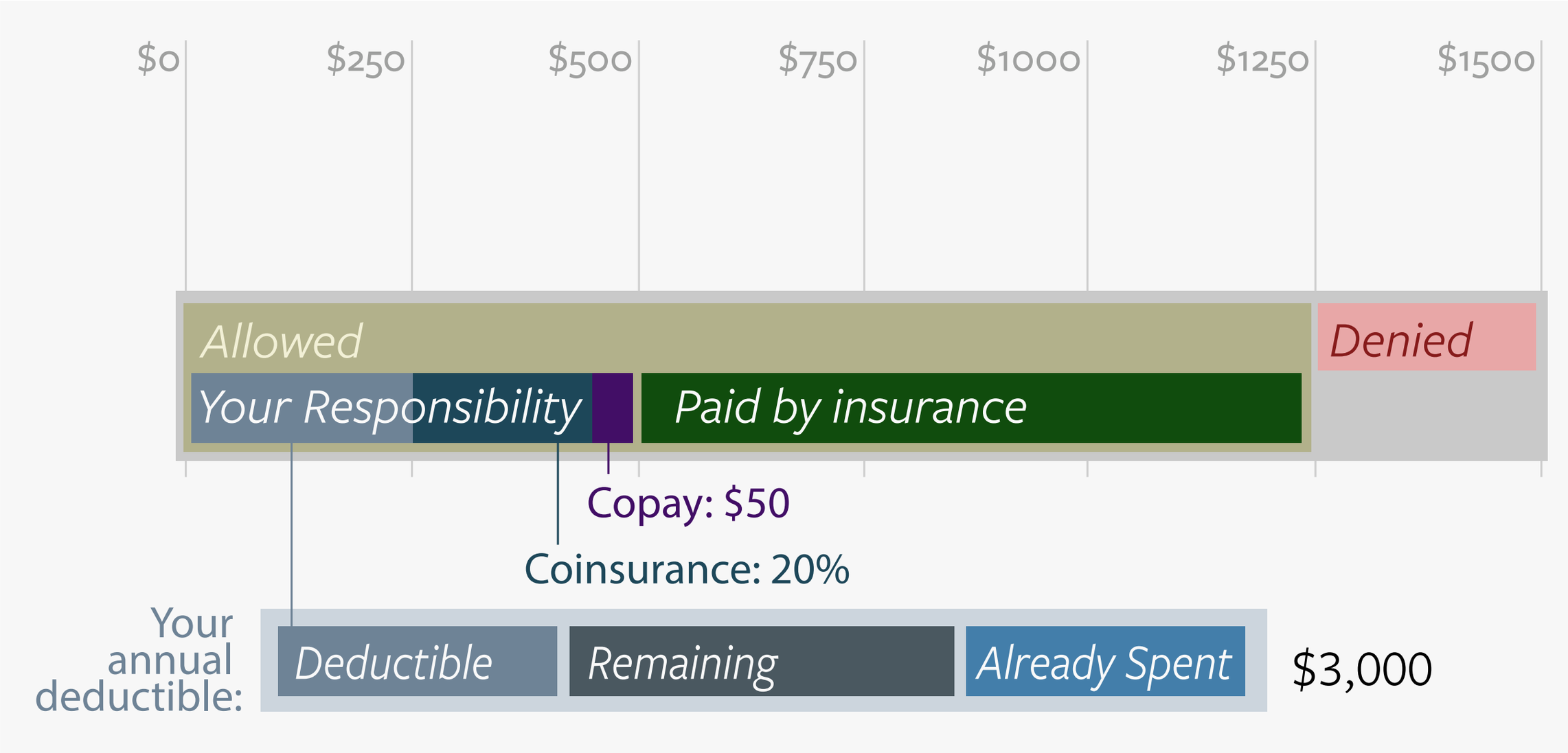
+ Denied

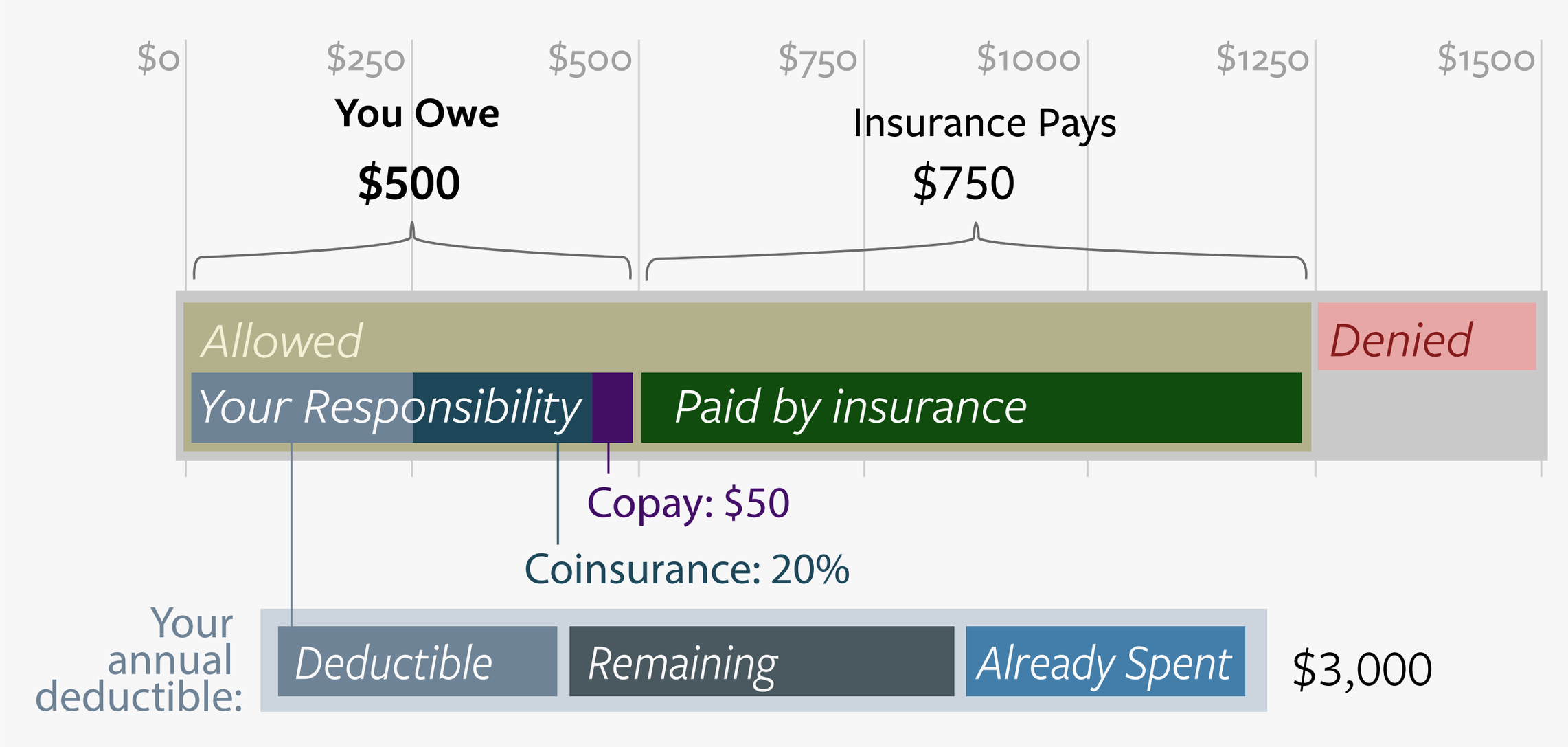
How do insurers compute patient responsibility?


$0
$250
$500
$750
$1000
$1250
$1500
Allowed
Denied


$0
$250
$500
$750
$1000
$1250
$1500
Allowed
Denied
Your annual deductible:
Deductible
Remaining
Already Spent
$3,000


$0
$250
$500
$750
$1000
$1250
$1500
Allowed
Denied
Copay: $50
Coinsurance: 20%
Your annual deductible:
Deductible
Remaining
Already Spent
$3,000


$0
$250
$500
$750
$1000
$1250
$1500
Allowed
Denied
Your Responsibility
Paid by insurance
Copay: $50
Coinsurance: 20%
Your annual deductible:
Deductible
Remaining
Already Spent
$3,000


$0
$250
$500
$750
$1000
$1250
$1500
You Owe
$500
Insurance Pays
$750
Allowed
Denied
Your Responsibility
Paid by insurance
Copay: $50
Coinsurance: 20%
Your annual deductible:
Deductible
Remaining
Already Spent
$3,000

Why Estimate?

- Answer “What does it cost?”
- Provide transparency into billing
- Allow patients to decide whether to use their insurance benefits

Demo

How Estimate?

Step:

Benefits Research → **Allowed Prediction** → **Estimate Prep** → **Estimate Display**

- Retrieve Deductible, Coinsurance from Insurer
- Filter benefits for relevance

- Predict expected reimbursment using historical claims
- Evaluate patient's family history against insurer medical policy

- Apply billing rules and policy

- Determine alerts based on current state
- Allow choice of bill type

Estimation Process

Step:	**Estimation Overall**	**Benefits Research** →	**Allowed Prediction** →	**Estimate Prep** →	**Estimate Display**
Key Classes:	`PriceCalculator Request (M)`	`Eligibility Inquiry (M)`	`CPTPricing Estimate (M)`	`Pricing Estimate (M)`	`Patient Pricing Estimate (C)`
Key Patterns and anti-Patterns:	Request/ Response Models	Mutability Mutable-until	Sub-App/Service	Rules Engine	Facade Logical Templates Rules Engine
Key APIs and interactions:	Billing engine Ordering API	Eligible API	Prediction App	Billing rules and policies	**Estimate JSON** **JS Client** `PEView`

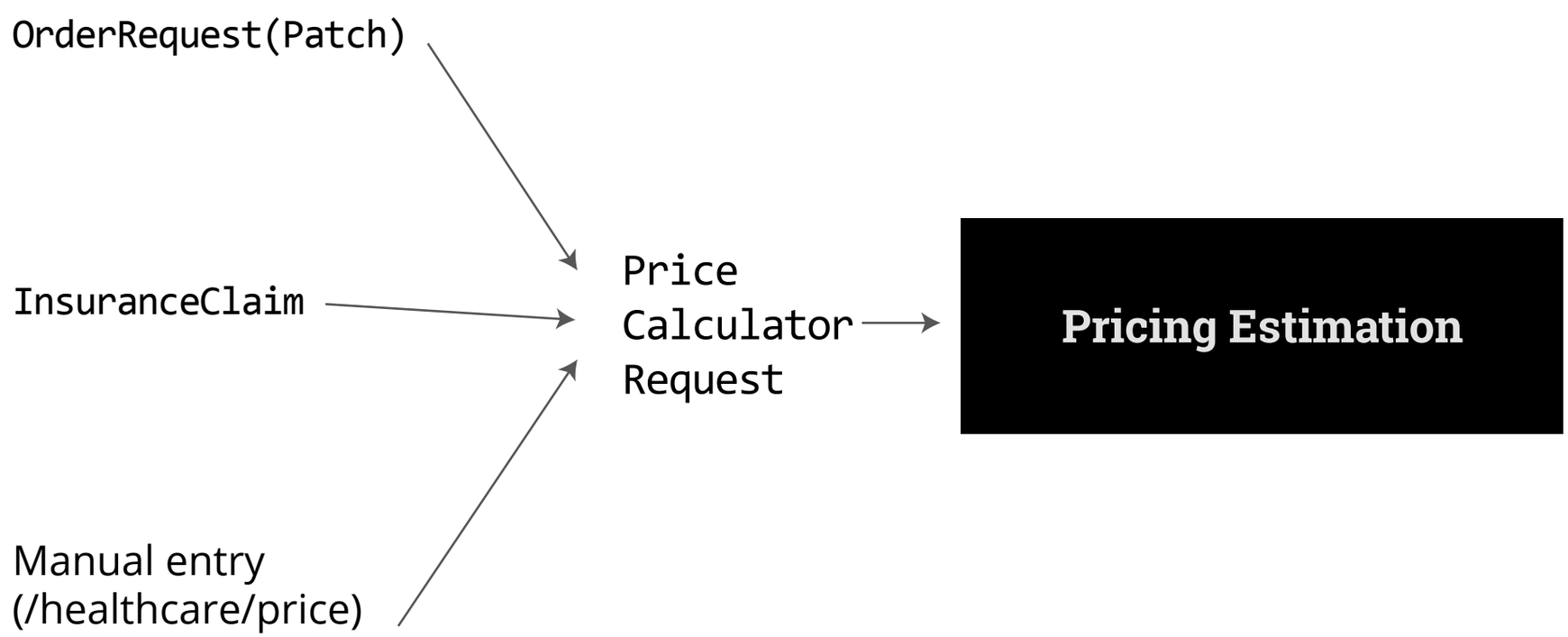
Key Design Constraints

- Edge cases dominate:
  - Significant percentage of patients don’t see a numeric estimate
- Recapitulates essentially all of billing
- Save everything patient sees
- Estimates change over time
- Desire for isolation (aspirational)

Estimation Process and Models

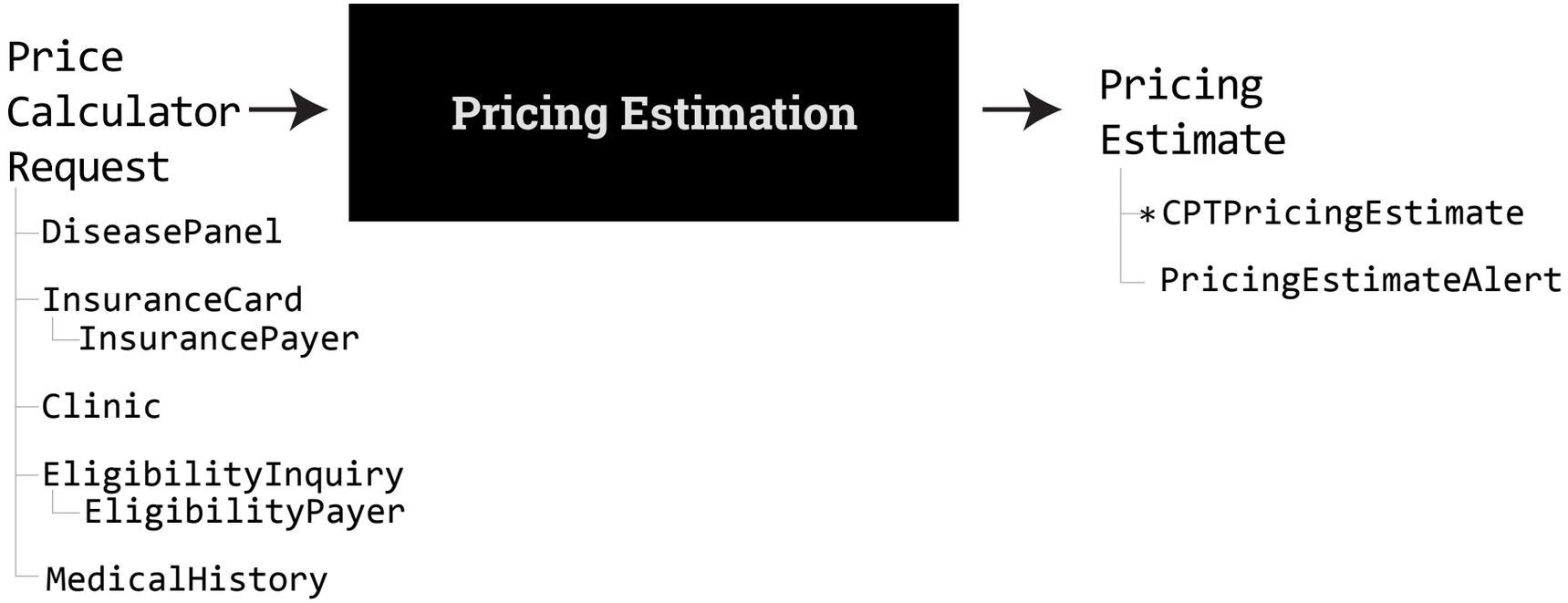
Request/Response

Estimates Input



```
class PriceCalculatorRequest(models.Model):
    first_name = HIPAAIdentifierCharField(max_length=100)
    last_name = HIPAAIdentifierCharField(max_length=100)
    gender = models.CharField(max_length=10, choices=G.GENDER_CHOICES)
    dob = HIPAAIdentifierDateField()
    state = models.CharField(choices=G.STATE_CHOICES)

    product = models.ForeignKey(Product, blank=True, null=True)
    disease_panel = models.ForeignKey(DiseasePanel, blank=True, null=True)

    payer = models.ForeignKey(InsurancePayer)
    in_network = models.NullBooleanField()
    member_id = HIPAAIdentifierCharField(max_length=100)
    group_number = models.CharField(max_length=100, blank=True)
    relationship_to_insured = models.CharField(
        choices=((G.RELATIONSHIP_SELF, 'My own plan'),
                 (G.RELATIONSHIP_DEPENDENT, 'Someone else\'s plan’)))

    physician = models.CharField(max_length=100, blank=True, default='')
    physician_state = models.CharField(choices=G.STATE_CHOICES)
    clinic = models.ForeignKey(Clinic, blank=True, null=True)

    timestamp = models.DateTimeField(auto_now_add=True)
    eligibility_inquiries = models.ManyToManyField(EligibilityInquiry)

    pricing_estimate = models.ForeignKey(
        PricingEstimate,
        blank=True,
        null=True)
    completed_on = models.DateTimeField(null=True)
    …
```

Request/Response



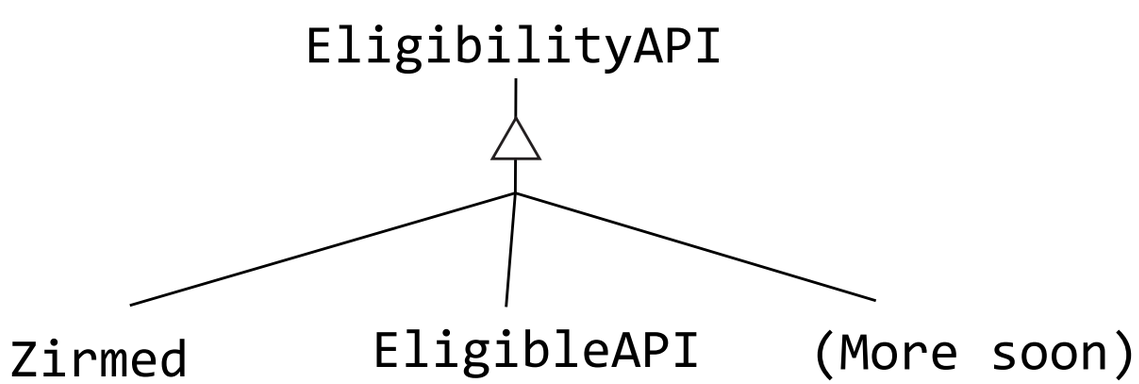
Benefits Research

Contact Insurer via Clearinghouse → Apply Coverage Filters → Apply Coverage Amount Rules

Electronic Data Interchange (EDI) standard to request benefits data. Clearinghouse reformats as JSON.

Exclude non-pertinent benefits amounts.

Replace known-bad data with fixed values.



EligibilityInquiry

```
request:JSONField
raw_response:(JSON)
timestamp_submitted
timestamp_responded
eligibility_payer
has_error
...
```

Example Inquiry Response

EligibleAPI


```
{
    "created_at": "2017-03-10T18:30:59-05:00",
    "demographics": {
        "dependent": {},
        "subscriber": {
            "address": {
                "city": "Lucedale",
                "state": "MS",
                "street_line_1": "9405 Deer Way",
                "street_line_2": null,
                "zip": "39452"
            },
            "dob": "1987-12-02",
            "first_name": "Eugenia",
            "gender": "F",
            "group_id": "00605103",
            "group_name": "BayCare Health System, Inc.",
            "last_name": "Clemmons",
            "member_id": "4035592912006100",
            "middle_name": "M"
        }
    },
    "eligible_id": "VFEYS4WOKSRPT",
    "insurance": {
        "contacts": [ ⋯ ],
        "id": "00001",
        "name": "Cigna HealthCare",
        "payer_type": "PR",
        "payer_type_label": "Payer",
        "service_providers": {
            "physicians": []
        }
    },
```

```
    "plan": {
        "group_name": "BayCare Health System, Inc.",
        "plan_name": "PPO",
        "plan_number": null,
        "plan_type": "PR",
        "plan_type_label": "Preferred Provider Organization (PPO)",
        "dates": [ ⋯ ],
        "exclusions": { ⋯ },
        "financials": {
            "coinsurance": { ⋯ },
            "copayment": { ⋯ },
            "cost_containment": { ⋯ },
            "deductible": {
                "remainings": {
                    "in_network": [
                        {
                            "amount": "500.00",
                            "authorization_required": null,
                            "comments": [
                                "Includes services provided by Client Specific
                                "For primary customers covered under single co
                                "For primary customers covered under family co
                                "Accumulators are shared between Medical AND V
                            ],
                            "contact_details": [],
                            "dates": [],
                            "description": null,
                            "insurance_type": null,
                            "insurance_type_label": null,
                            "level": "FAMILY",
```

Example Inquiry Response

Coinsurance

10% type: Diagnostic Lab (5) level: INDIVIDUAL pos: (none)
comments: EPO NETWORK

20% type: Diagnostic Lab (5) level: INDIVIDUAL pos: (none)
comments: PAR NETWORK
filtered: (not filtered)

10% type: Diagnostic Lab (5) level: INDIVIDUAL pos: Office (11)
comments: EPO NETWORK

20% type: Diagnostic Lab (5) level: INDIVIDUAL pos: Office (11)
comments: PAR NETWORK

Filtered Inquiry Response

Coinsurance

20%

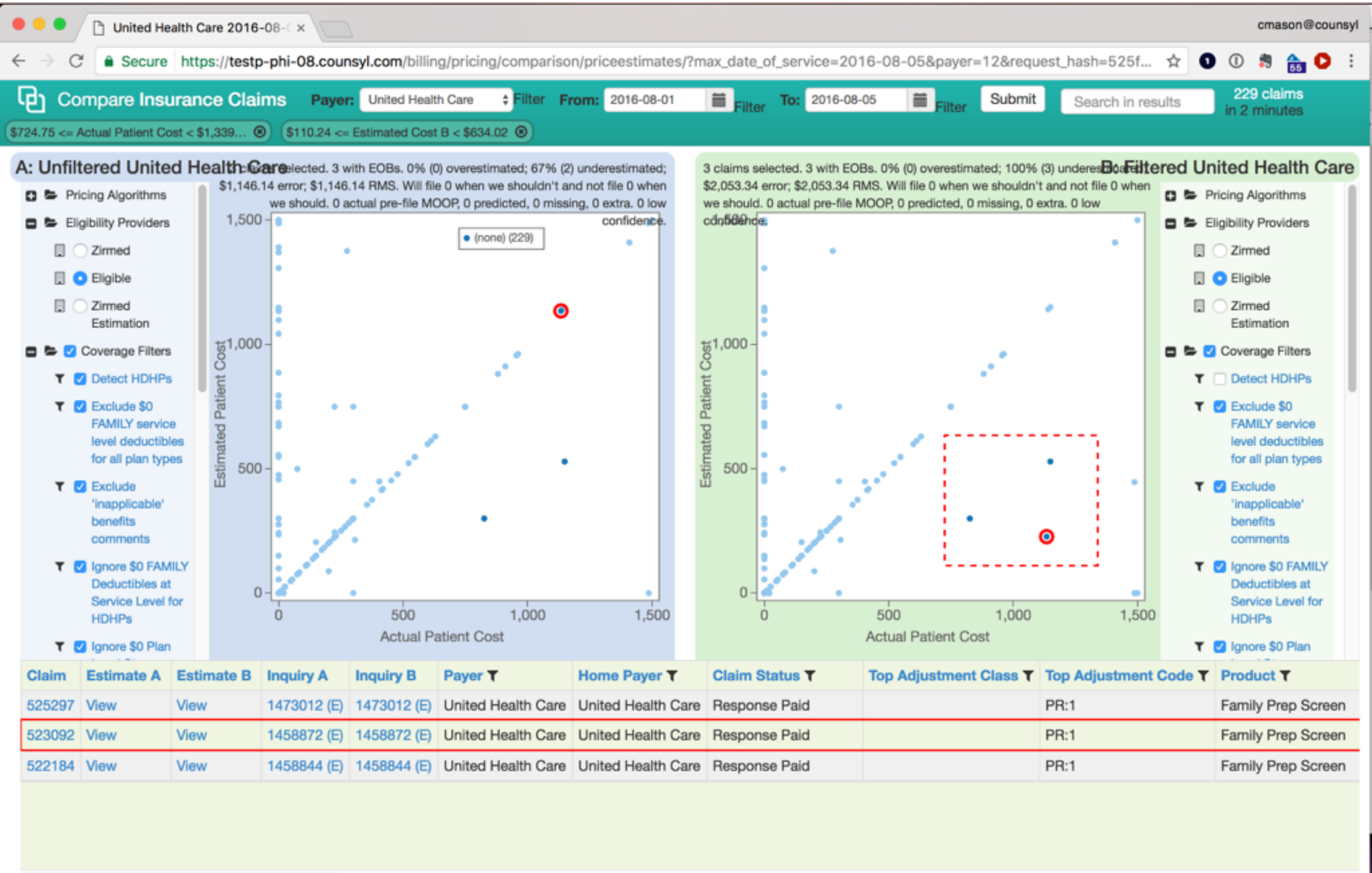
10% type: Diagnostic Lab (5) level: INDIVIDUAL pos: (none)
comments: EPO NETWORK
filtered: Out By: Ignore EPO NETWORK

20% type: Diagnostic Lab (5) level: INDIVIDUAL pos: (none)
comments: PAR NETWORK
filtered: (not filtered)

10% type: Diagnostic Lab (5) level: INDIVIDUAL pos: Office (11)
comments: EPO NETWORK
filtered: Out By: Ignore EPO NETWORK

20% type: Diagnostic Lab (5) level: INDIVIDUAL pos: Office (11)
comments: PAR NETWORK
filtered: Out By: Ignore POS:Office

Manual Creation of Filters



Claim	Estimate A	Estimate B	Inquiry A	Inquiry B	Payer	Home Payer	Claim Status	Top Adjustment Class	Top Adjustment Code	Product
525297	View	View	1473012 (E)	1473012 (E)	United Health Care	United Health Care	Response Paid		PR:1	Family Prep Screen
523092	View	View	1458872 (E)	1458872 (E)	United Health Care	United Health Care	Response Paid		PR:1	Family Prep Screen
522184	View	View	1458844 (E)	1458844 (E)	United Health Care	United Health Care	Response Paid		PR:1	Family Prep Screen

Anti-Pattern: **Masochistic Mutability**

Goal: If at all possible, make it immutable.

Alternatives: “Mutable-until”. Change Log. Patches.

Example:

InsuranceClaim

`raw_deductible:Decimal`
`remaining_deductible:Decimal`
`co_insurance:Decimal`
`...`

Pattern: **Mutable-until** (aka Immutable-after)

Goal: Model is mutable while being processed, but a final event marks it as immutable.

Alternatives: Changelog, Patches

Example:

EligibilityInquiry

`raw_response`
`timestamp_responded`
`...`

Allowed Prediction

Recall:

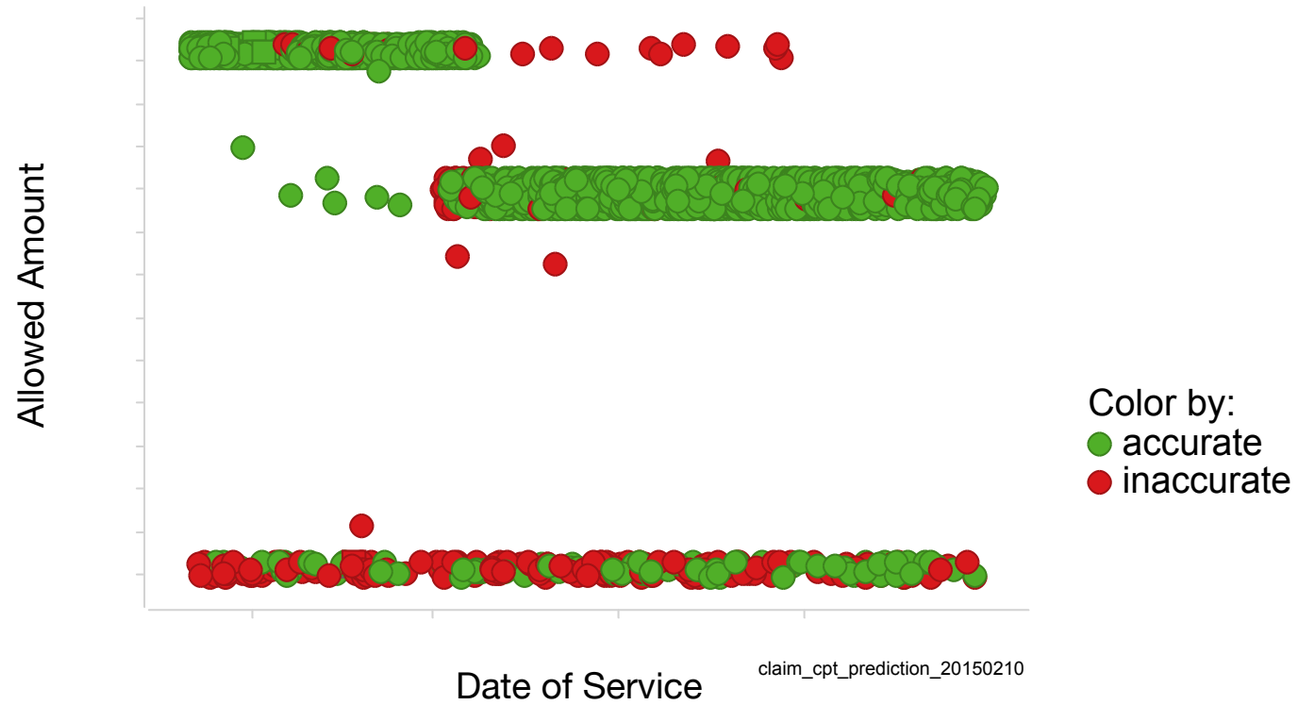
Allowed = Amount insurance thinks our test is worth

CPT: 81220

For Each CPT:

- Find similar historical claims
- Take most common allowed $

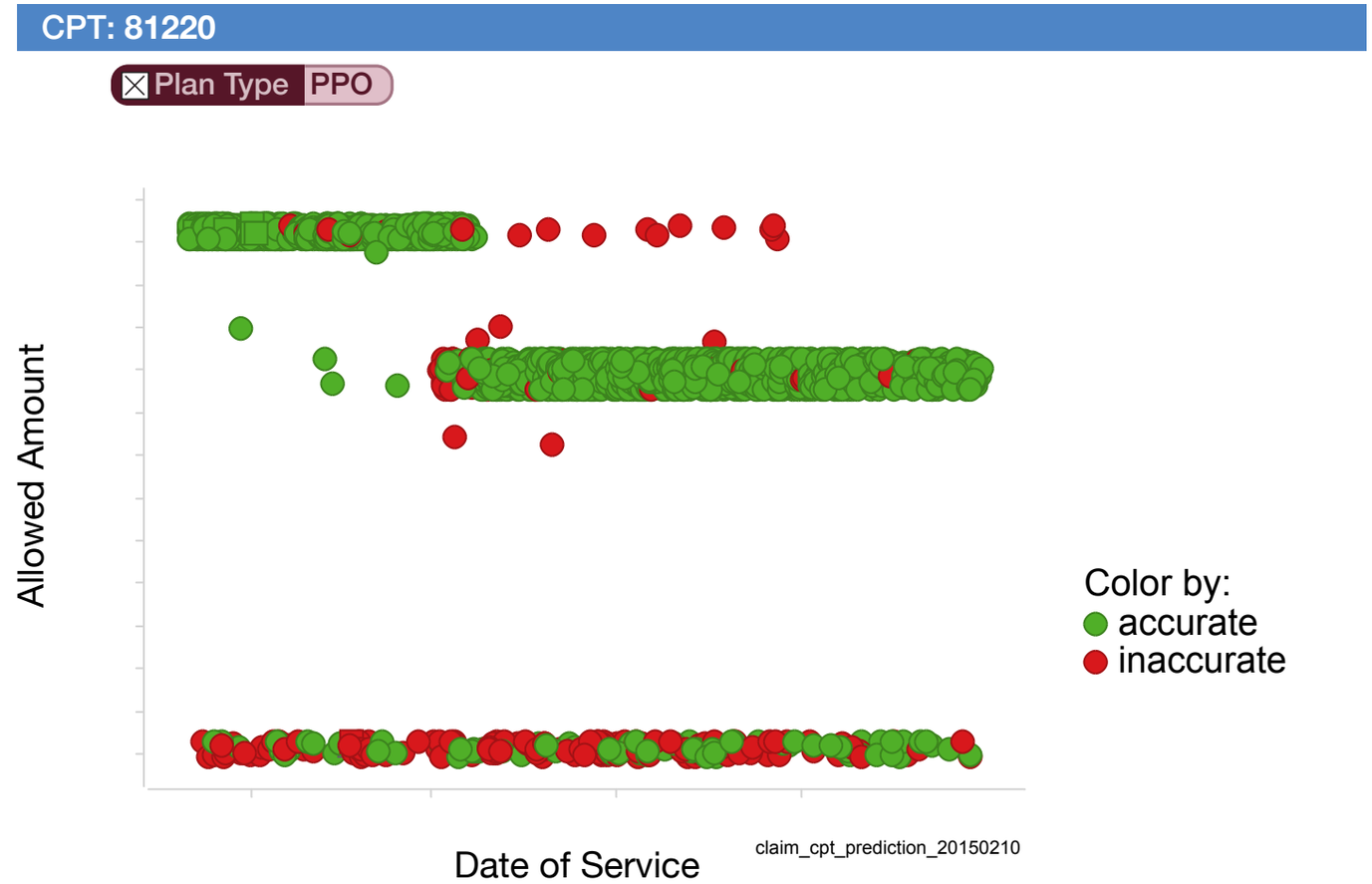
Sum allowed values across CPTs



For Each CPT:

- Find similar historical claims
- Take most common allowed $

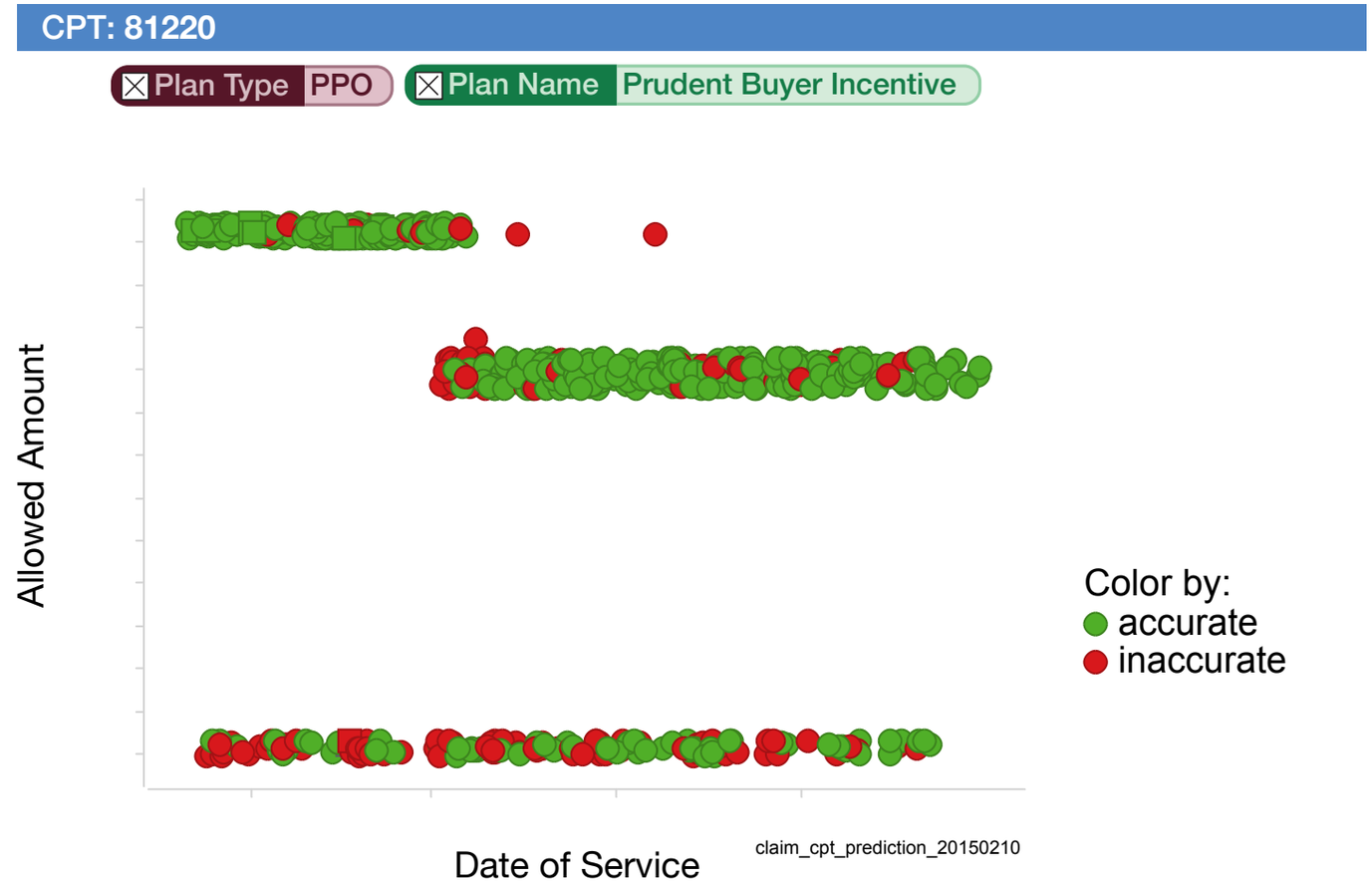
Sum allowed values across CPTs



For Each CPT:

- Find similar historical claims
- Take most common allowed $

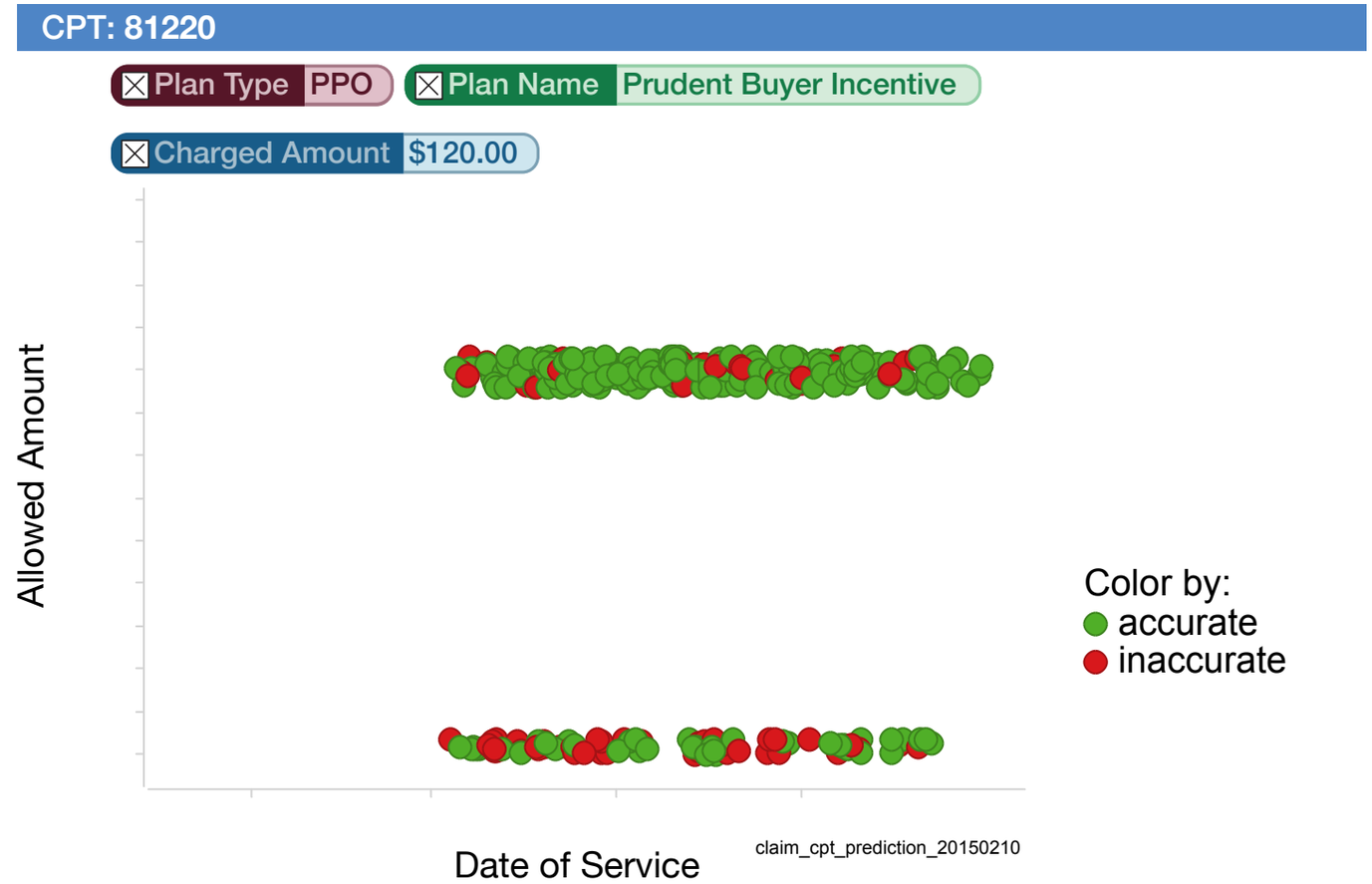
Sum allowed values across CPTs



For Each CPT:

- Find similar historical claims
- Take most common allowed $

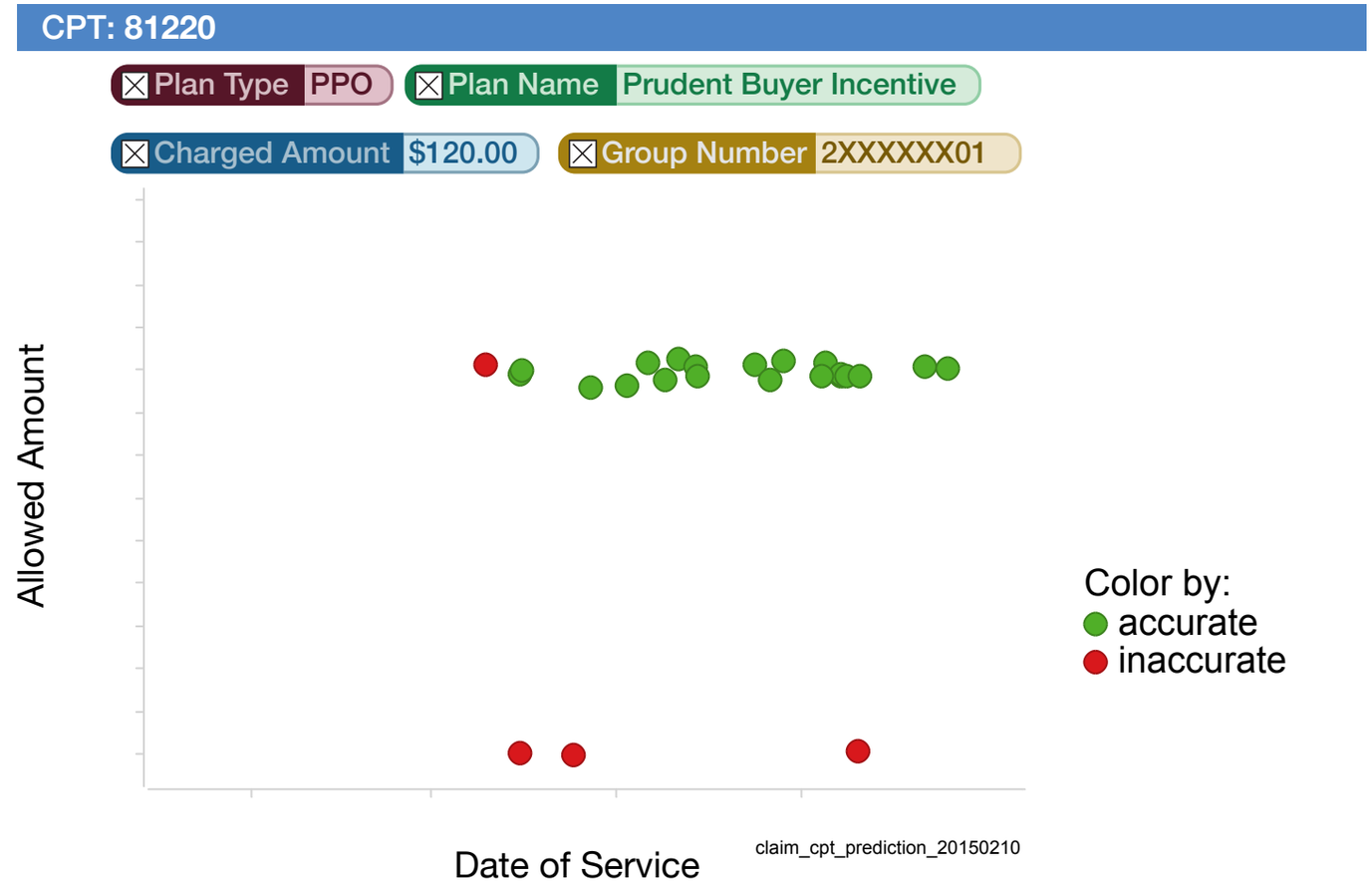
Sum allowed values across CPTs



For Each CPT:

- Find similar historical claims
- Take most common allowed $

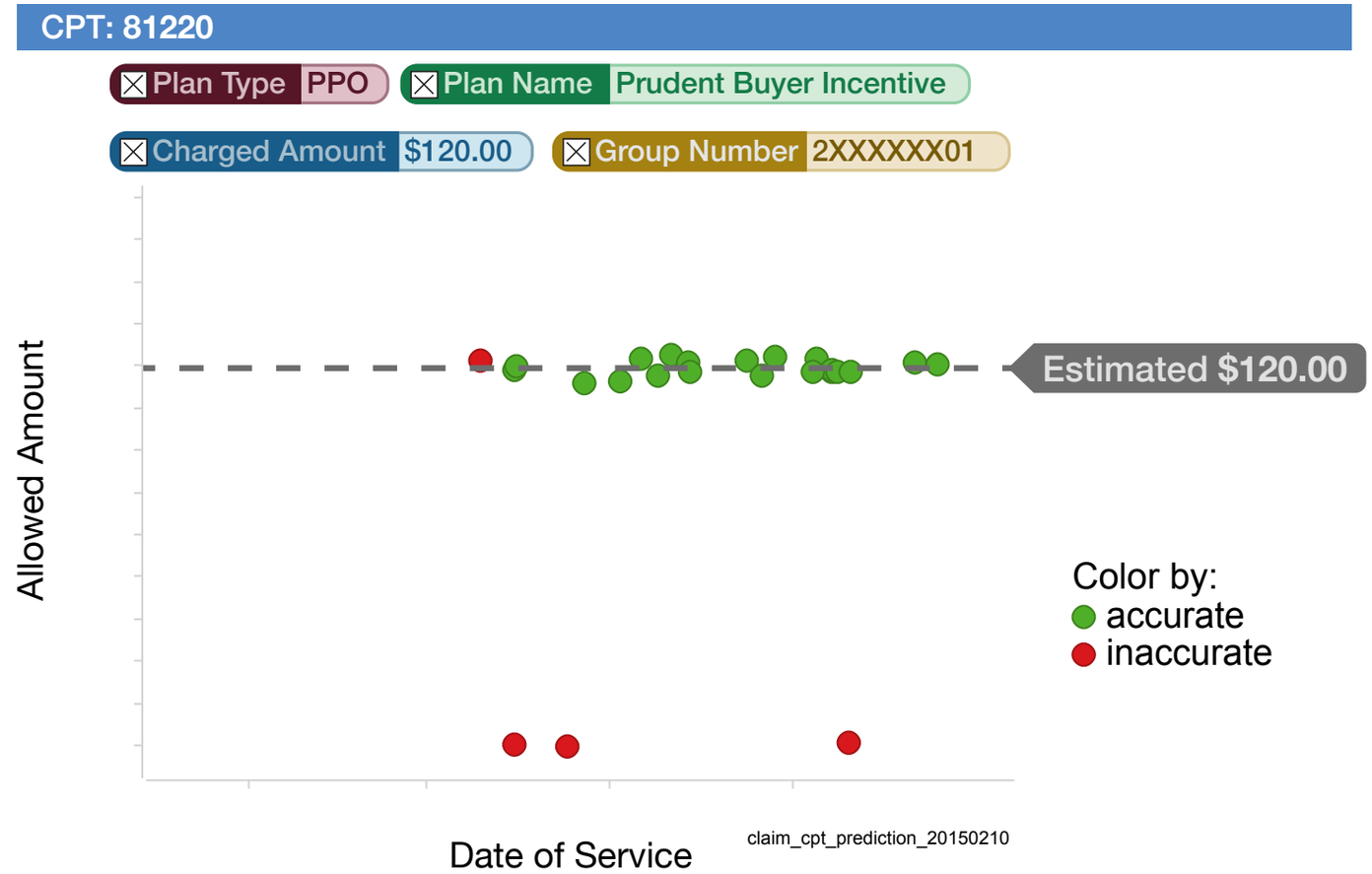
Sum allowed values across CPTs



For Each CPT:

- Find similar historical claims
- Take most common allowed $

Sum allowed values across CPTs



Pluggable Algorithms

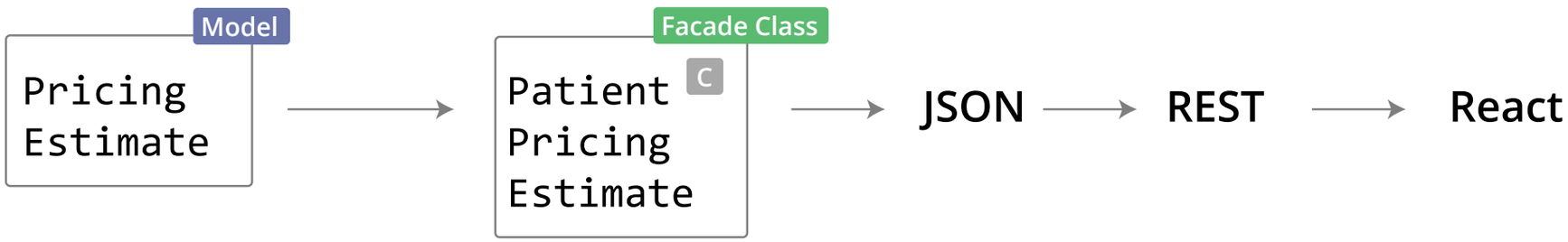
```
class PriceEstimator(object):
    """
    Interface for predicting PricingEstimates.
    """
    def get_algorithm_constant(self):  # pragma: no cover
        """
        Return PRICING_ALGORITHM constant for this algorithm.
        """
        raise NotImplementedError()

    def get_pricing_estimate(self,
                             price_calculator_request,
                             coverage_details, in_network,
                             requested_by, code_set=None,
                             disable_moop=False,
                             use_coverage_amounts_rules=True,
                             ):
        return PricingEstimate(…)
```

Estimate Prep

- Sum up per-CPT estimates
- Apply patient’s benefits
- Apply billing rules / policies
- Save PricingEstimate

```
class PricingEstimate(TaggableMixin, models.Model):
    eligibility_inquiry = models.ForeignKey(EligibilityInquiry)
    disease_panel = models.ForeignKey(DiseasePanel)
    code_set = models.ForeignKey('coding.CodeSet')
    algorithm = EnumField(EC.PRICING_ALGORITHM)
    git_hash = models.CharField()  # Git hash of code that generated estimate.
    timestamp = models.DateTimeField(auto_now_add=True)

    estimated_copayment = models.DecimalField(decimal_places=2, max_digits=20)
    estimated_deductible = models.DecimalField(decimal_places=2, max_digits=20)
    estimated_coinsurance = models.DecimalField(decimal_places=2, max_digits=20)
    estimated_coinsurance_percent = models.DecimalField(decimal_places=2, max_digits=20)
    estimated_other_patient_responsibility = models.DecimalField(decimal_places=2, max_digits=20)
    estimated_total_before_comp = models.DecimalField(decimal_places=2, max_digits=20)
    estimated_total_before_comp_lower_bound = models.DecimalField(decimal_places=2, max_digits=20)

    coverage_filters_set = models.ForeignKey(HistoricalCoverageAmountFilterSet,null=True)
    billing_policy = models.ForeignKey(BillingPolicy, null=True)

    confident_in_benefit_amounts = models.NullBooleanField(
        "Confident in benefit amounts",
        help_text=('Are we relatively confident in the coverage amounts '
                   'used to generate this estimate?'),
        default=None)

    confident_in_allowed_amounts = models.NullBooleanField(
        "Confident in allowed amounts",
        help_text=('Are we relatively confident in the estimated allowed '
                   'amount?'),
        default=None)
....
```

Estimate Display



“Edge” Cases

- **Bad patient info:** not able to id patient
- **Prior auth, Pre-test counseling:** insurance requirements to reduce demand
- **No benefits info:** able to find patient but no benefits returned.
- …

Anti-Pattern: **Logical Templates**

Problem: Embedding display logic and variable content in HTML templates.

Alternatives: Content Management System, Rules Engines

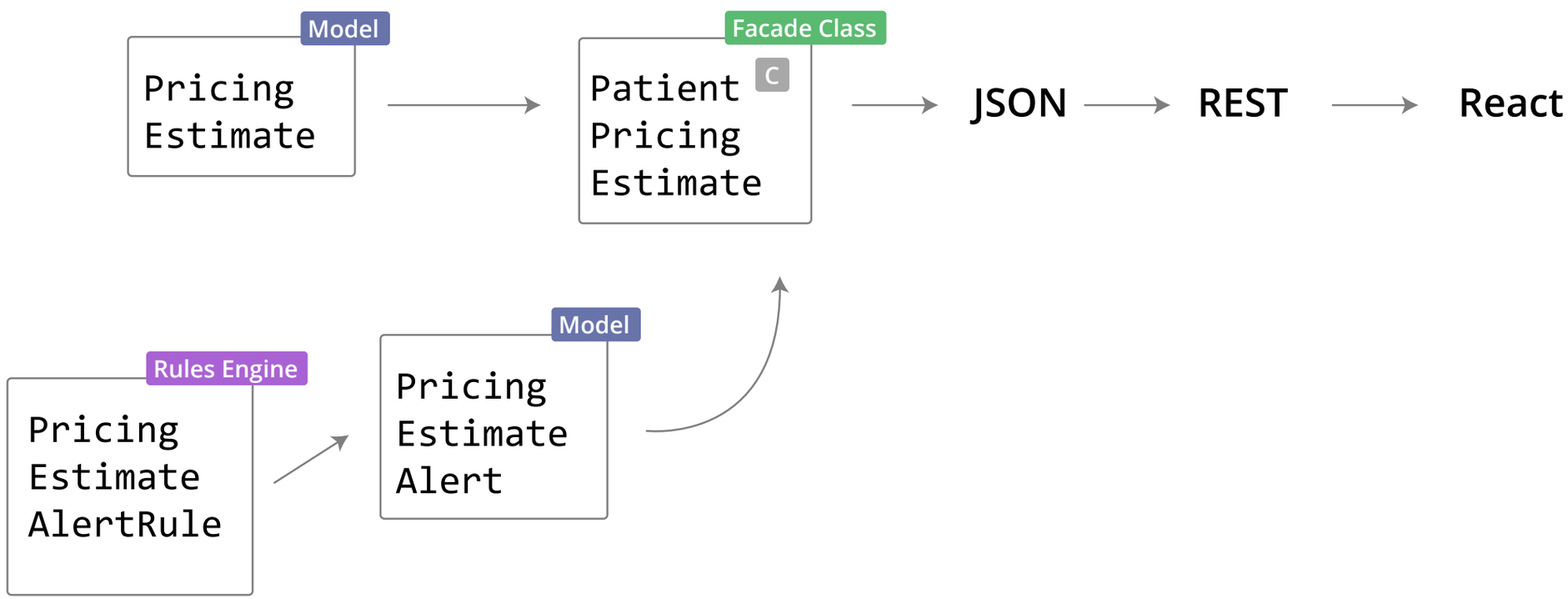
Example:

```
    {% if patient_estimate.show_prior_auth_in_description %}
      {% include "my/pricing_estimate/_needs_prior_auth_description.html" %}
    {% endif %}
    <div class="nav-links">
      {% if patient_estimate.should_show_breakdown_section %}
      <a href="#estimate-breakdown">See full breakdown &raquo;</a>
...
      {% if patient_estimate.should_show_order_now_and_edit_buttons %}
          <a class="btn btn btn--default price-calculator-start-over" href="#">Edit my information</a>
          <a class="btn btn btn--primary" href="{{ product_order_url }}">Order now</a>
      {% endif %}
    </div>
  </div>
...
 {% if patient_estimate.show_prior_auth_notice %}
   {% include "my/pricing_estimate/_needs_prior_auth_details.html" %}
 {% endif %}
```

Alerts

- “Mini-CMS” for “Unhappy Cases”
- Let’s business/insurance/UX experts change how we present non-numeric estimates.
- Markdown + Rules Engine

Estimate Display



Demo

Pattern: **Rules Engine**

Goal: Policy as data

Alternatives: Hard coded logic, BRMS, Expression Language

Example:

```
class PricingEstimateAlertRule(gradgrind.NoConflictCheckingRulesEngine):
    atom_model = 'brochure.PriceCalculatorRequest'

    uuid = models.UUIDField()
    description = models.CharField()
    priority = models.IntegerField()

    # Value Fields:
    add_alerts = models.ManyToManyField(PricingEstimateAlert)
    remove_alerts = models.ManyToManyField(PricingEstimateAlert)

    # Predicate matching fields:
    products = models.ManyToManyField('order.Product', blank=True)
    payers = models.ManyToManyField('billing.InsurancePayer', blank=True)
    ...
```

```
predicates = Predicates(
    IntersectionPredicate('product', 'products'),
    IntersectionPredicate('payer', 'payers'),
    IntersectionPredicate(
        'insurance_claim__code_set__service_lines_
        'cpt_codes'),
    FieldPredicate('in_network'),
    FieldPredicate(
        'has_bad_patient_info',
        field_description='Has Bad Patient Info'),
    ...
```

Select pricing estimate alert rule to change

Import Export Add p

ID	Description	Priority	Active	Rule summary	Add Alerts	Remove Alerts
6	Bad Patient Info	4	✓	▪ **Has Bad Patient Info:** True ▪ **Has Order:** True	▪ bad-patient-info	▪ prior-auth-text ▪ pre-test-counseling-text ▪ low-confidence-text

Please update your information

We do not have your correct information on file and cannot provide you with an estimate of your out-of-pocket cost with your insurance.

If you want to pay with insurance, please update your information. You can do so online below or by calling us.

If your information is not updated, or you do not qualify for insurance coverage, you will receive our cash price of $349.00.

15	Prior Auth Required Aetna ICS	1	✓	▪ **Product:** Inherited Cancer Screen ▪ **Insurance Payer:** Aetna ▪ **Needs Prior Auth:** True	▪ prior-auth-text ▪ prior-auth-required-aetna-ics	▪ prior-auth-required

Data Design / Modeling

Data is forever

- **Schemas can change**, but essential content must **remain meaningful** essentially forever.
- Take the long view: design flexibly.
- Consider impact of schema changes on:
  - Data analysis
  - Customer support
  - Auditing/compliance

Design for test

- Ability to anonymize
- Consider how heavyweight the fixtures
- Example: alerts

Testing Scenarios

```yaml
fps_seq_not_confident_in_benefits:
  type: Low confidence in benefits
  product_abbrv: Foresight
  description: >
    "Low confidence" estimate: Foresight sequencing claim where we don't get any
    benefits information back from insurer. Different language is displayed,
    and estimate uses worst case values assuming 100% cost sharing.
  prod_claim_query: >
    InsuranceClaim.objects.filter(
      status=G.CLAIM_RESPONSE_PAID,
      order__product__slug='family-prep-screen',
      # Specifically low confidence in benefits:
      pricing_estimate__confident_in_benefit_amounts=False,
      pricing_estimate__confident_in_allowed_amounts=True,
      # Avoid default benefits claims, as these are handled by a separate scenario.
      bypassed_eligibility_lookup=False,
    )
  requires_prod_data_from:
    claim: 596702  # BCBS MD claim with no benefits values.
  pre_actions:
    - load_anonymized_test_objects: 'claim_596702'
  pre_asserts:
    claim:
      total_charges: Decimal('123.04')
  claim: $claim_596702
  eligibility_inquiry: $eligibility_inquiry_1583594
  prediction_engine: $prediction_engine_claim_596702
  post_actions:
    - set_claim_state: G.CLAIM_RESEARCHED
  post_asserts:
    claim:
      cash_price: Decimal(‘345.67')
    pricing_estimate:
      confident_in_allowed_amounts: true
      confident_in_benefit_amounts: false
      estimated_allowed_amount: Decimal(‘456.78')
```

Internal Tooling

- Tool for A/B comparison of algorithm output on a given claim
- Internal view of eligibility data
- Alert import/export
- Missing: good internal estimate UI

Conclusions

- ML only a tiny part of solution
- Edge cases dominate
- Sometimes it revisiting many times to get to a solid solution
- We’re hiring!
- ROBOTS!