Example Scenario: Personalized Health Insurance Recommendations

User Profile

- **Age:** 35
- **Location:** Melbourne
- **Health Needs:** Maternity coverage, no waiting periods.
- **Budget:** Prefers policies under $150/month.
- **Additional Preferences:** Prefers insurers with high customer satisfaction ratings.

Step-by-Step Breakdown

1. Data Collection

- **User Data:**
  - Age: 35
  - Location: Melbourne (encoded as a categorical feature, e.g., "VIC").
  - Health Needs: Maternity coverage (encoded as a binary feature: 1 for maternity, 0 otherwise).
  - Budget: $150/month (normalized to a scale of 0–1, e.g., 0.75).
  - Customer Satisfaction Preference: High (encoded as a binary feature: 1 for high, 0 for low).
- **Policy Data:**
  - **Policy X:**
    - Premium: $120/month (normalized to 0.6).
    - Waiting Period: 0 days for maternity (encoded as 0).
    - Coverage: Maternity, dental, optical.
    - Insurer Rating: 4.5/5 (normalized to 0.9).
  - **Policy Y:**
    - Premium: $135/month (normalized to 0.675).
    - Waiting Period: 30 days for maternity (encoded as 30).
    - Coverage: Maternity, neonatal care, hospital.
    - Insurer Rating: 4.2/5 (normalized to 0.84).
  - **Policy Z:**
    - Premium: $110/month (normalized to 0.55).
    - Waiting Period: 0 days for maternity (encoded as 0).
    - Coverage: Maternity, partnered with Royal Women’s Hospital.
    - Insurer Rating: 4.7/5 (normalized to 0.94).
- **Interaction Data:**

- Historical data showing that users with similar profiles (e.g., age 30–40, maternity needs) often prefer policies with no waiting periods and high insurer ratings.

---

2. Model Input

The user's features are preprocessed and fed into the **User Tower** of the TFRS model:

python
Copy
```
user_input = {
  "age": 0.35,  # Normalized (35 / 100)
  "location": "VIC",  # Categorical (Victoria)
  "maternity_coverage": 1,  # Binary (1 = needed)
  "budget": 0.75,  # Normalized ($150/month)
  "insurer_rating_preference": 1  # Binary (1 = high rating preferred)
}
```

The **Policy Tower** processes the policy features:

python
Copy
```
policy_dataset = [
  {
    "premium": 0.6,
    "waiting_period": 0,
    "coverage": ["maternity", "dental", "optical"],
    "insurer_rating": 0.9
  },
  {
    "premium": 0.675,
    "waiting_period": 30,
    "coverage": ["maternity", "neonatal", "hospital"],
    "insurer_rating": 0.84
  },
  {
    "premium": 0.55,
    "waiting_period": 0,
    "coverage": ["maternity", "hospital_partnership"],
    "insurer_rating": 0.94
  }
]
```

---

3. Model Inference

The TFRS model computes embeddings for the user and policies, then calculates similarity scores:

python
Copy

```
# User embedding
user_embedding = model.user_model(user_input)

# Policy embeddings
policy_embeddings = model.policy_model(policy_dataset)

# Similarity scores (dot product)
scores = tf.matmul(user_embedding, policy_embeddings, transpose_b=True)
```

The scores might look like this:

- Policy X: 0.92
- Policy Z: 0.89
- Policy Y: 0.75

---

4. Top-K Recommendations

The model retrieves the top-2 policies based on similarity scores:

python
Copy
```
top_k_indices = tf.math.top_k(scores, k=2).indices.numpy()
recommended_policies = [policy_dataset[i] for i in top_k_indices]
```

Output:

1. **Policy X:**
   - Premium: $120/month
   - Waiting Period: 0 days for maternity
   - Coverage: Maternity, dental, optical
   - Insurer Rating: 4.5/5
2. **Policy Z:**
   - Premium: $110/month
   - Waiting Period: 0 days for maternity
   - Coverage: Maternity, partnered with Royal Women's Hospital
   - Insurer Rating: 4.7/5

---

5. Explanation for Recommendations

The platform provides explanations for why these policies were recommended:

- **Policy X:**
  - "Recommended because it has no waiting period for maternity and fits your budget."
- **Policy Z:**

    - "Recommended because it partners with Royal Women's Hospital and has a high insurer rating."

---

6. User Interaction

The user can:

- Click on a policy to view detailed coverage and exclusions.
- Adjust filters (e.g., increase budget to $160/month) to see updated recommendations.
- Compare policies side-by-side.

Technical Implementation Details

Model Training

- **Loss Function:**
  Use a retrieval task with factorized top-K metrics to optimize for accurate policy recommendations.

  python

  Copy

  ```
  task = tfrs.tasks.Retrieval(
    metrics=tfrs.metrics.FactorizedTopK(
      candidates=policy_dataset.batch(128).map(policy_model)
    )
  )
  ```
- **Training Data:**
  Use historical user-policy interactions (e.g., clicks, purchases) to train the model.
  Example:

  python

  Copy

  ```
  train_data = tf.data.Dataset.from_tensor_slices({
    "user_features": user_data,
    "policy_features": policy_data,
    "interaction": interaction_labels  # 1 (clicked/purchased) or 0 (ignored)
  })
  ```

Deployment

- **API Endpoint:**
  Deploy the model using **TensorFlow Serving** or **FastAPI** to serve recommendations in real time.
  Example API call:

json

Copy

```
POST /recommendations
{
  "user_id": "123",
  "features": {
    "age": 35,
    "location": "VIC",
    "maternity_coverage": 1,
    "budget": 150,
    "insurer_rating_preference": 1
  }
}
```

- **Response:**

json

Copy

```
{
  "recommended_policies": [
    {
      "policy_name": "Policy X",
      "premium": 120,
      "waiting_period": 0,
      "coverage": ["maternity", "dental", "optical"],
      "insurer_rating": 4.5
    },
    {
      "policy_name": "Policy Z",
      "premium": 110,
      "waiting_period": 0,
      "coverage": ["maternity", "hospital_partnership"],
      "insurer_rating": 4.7
    }
  ]
}
```

Business Impact

- **Increased Conversions:** Personalized recommendations lead to higher engagement and purchase rates.
- **Improved User Satisfaction:** Users find policies tailored to their needs, reducing decision fatigue.

- **Competitive Advantage:** Differentiates the platform from competitors using rule-based or non-AI systems.

This detailed example demonstrates how TensorFlow Recommenders can be used to deliver **personalized, data-driven health insurance recommendations** in the Australian market.