## GhostCox Thesis Project: Summary and Plan

This document summarizes the discussion, decisions, and plan for the MSc thesis project focused on interpreting non-linear survival models using the Ghost Variables methodology ("GhostCox").

**1. Thesis Aim & Context**

* **Goal:** To develop and evaluate the "GhostCox" method for interpreting non-linear survival models of the form h(t, \mathbf{x}\_i) = h\_0(t) \exp(f(\mathbf{x}\_i)), where f(\mathbf{x}) is estimated flexibly (e.g., using Machine Learning, GAMs).
* **Objective:** Apply the Ghost Variables methodology (Delicado & Peña, 2023) to assess the relevance of individual predictor variables (X\_j) to the estimated non-linear risk function \hat{f}(\mathbf{x}). This provides an interpretable measure of variable importance complementary to effect-size measures like Hazard Ratios.
* **Motivation:** Address the interpretability challenge ("black box" problem) of complex survival models, building on the need identified by works like Sundrani & Lu (2021) but using a different technique (Ghost Variables instead of SHAP).

**2. Relevant Literature Reviewed**

* **Delicado & Peña (2023):** Introduced Ghost Variables (*\hat{X}j = \mathbb{E}(X\_j | \mathbf{X}{-j})*) and the relevance measure RV\_{gh}(X\_j) based on prediction changes when substituting X\_j with \hat{X}\_j. Also introduced the Relevance Matrix \mathbf{V}. Used MSPE for normalization in regression.
* **Sundrani & Lu (2021):** Developed a method to compute HRs from ML survival models (XGBoost) using SHAP values, enhancing interpretability for clinicians.
* **Bednarski (1993):** Proposed robust Cox estimators using Fréchet differentiable functionals via weighted score equations to handle model deviations.
* **Farcomeni & Viviani (2011):** Proposed a robust Cox estimator based on trimming observations with low partial likelihood contributions to handle outliers.
* **Ortiz & Becerra (2024):** (Example thesis) Provided context on robust statistics, outlier detection, and projection pursuit methods (like Stahel-Donoho).

**3. Proposed GhostCox Methodology**

* **Model Structure:** h(t, \mathbf{x}\_i) = h\_0(t) \exp(f(\mathbf{x}\_i)).
* **Stage 1: Fitting \boldsymbol{\hat{f}(\mathbf{x})}:** Estimate the non-linear function f(\mathbf{x}) using a suitable survival model (e.g., Random Survival Forest - RSF) on training data (X\_{\text{train}}, y\_{\text{train}}).
* **Stage 2: Interpretation via Ghosts (on Test Set):**
  + Estimate ghost variables *\hat{\mathbf{x}}j = \hat{\mathbb{E}}(X\_j | \mathbf{X}{-j, \text{test}})* for each X\_j using a regression model (e.g., Random Forest Regressor - RFR) fitted on test covariates X\_{\text{test}}.
  + Calculate original risk scores *\hat{f}{\text{test}} = \hat{f}(X{\text{test}})* and ghost-substituted scores *\hat{f}{\text{test}, \hat{j}} = \hat{f}(X{\text{test}, \hat{j}})*.
  + Calculate relevance numerator: R\_j^{\text{num}} = \frac{1}{n\_{\text{test}}} \sum\_{i \in \text{test}} (\hat{f}(x\_i) - \hat{f}(x\_{i, \hat{j}}))^2.
  + Calculate normalization factor: *Var(\hat{f}{\text{test}}) = \text{Variance of } \hat{f}{\text{test}}*.
  + Calculate final relevance: RV\_{gh}(X\_j) = R\_j^{\text{num}} / Var(\hat{f}\_{\text{test}}).
  + (Optional) Calculate Relevance Matrix: \mathbf{V} = \frac{1}{n\_{\text{test}}} \frac{\mathbf{A}'\mathbf{A}}{Var(\hat{f}\_{\text{test}})}, where \mathbf{A} contains prediction changes.
* **Justification for Normalization:** Var(\hat{f}\_{\text{test}}) is used as a practical and conceptually analogous measure to MSPE (used by Delicado & Peña) for the risk score output, interpreting RV\_{gh}(X\_j) as the proportion of risk score variance uniquely attributable to X\_j.

**4. Python Implementation Structure**

* **SurvivalDataGenerator**: Generates simulated data with options for linear/non-linear predictors, correlated/uncorrelated features, censoring, and contamination. (Corrected to use feature\_generation\_type and corr\_matrix arguments).
* **NonLinearSurvivalModel**: Wrapper for scikit-survival models (initially RSF). Handles fitting and predicting risk scores \hat{f}(\mathbf{x}).
* **GhostVariableEstimator**: Estimates ghost variables \hat{\mathbb{E}}(X\_j | \mathbf{X}\_{-j}) using specified regressors (RF, LM, GAM).
* **GhostCoxInterpreter**: Orchestrates Stage 2 interpretation, calling the survival model and ghost estimator to calculate RV\_{gh} and optionally \mathbf{V}. (Updated to include normalization).
* **ExperimentRunner**: Manages running simulation scenarios, collecting results (C-index, relevance, ranks, timing), and generating summary tables/plots.

**5. Environment & Code Issues Encountered & Resolved**

* Initial ValueError: numpy.dtype size changed... due to binary incompatibility between NumPy and SciPy/Seaborn. Resolved by creating a virtual environment (venv) and reinstalling packages (pip install --no-cache-dir ...).
* Initial TypeError: ... got multiple values for argument 'n\_features' when calling feature generation functions via partial within SurvivalDataGenerator. Resolved by modifying SurvivalDataGenerator to use explicit type arguments (feature\_generation\_type, corr\_matrix) and directly calling the appropriate global helper functions (generate\_uncorrelated\_features, generate\_correlated\_features) from within \_generate\_features.

**6. Simulation Study 1 Design**

* **Goal:** Verify GhostCox variable ranking.
* **Model:** RSF survival model, RF ghost estimator.
* **Parameters:** n=500, p=5, ~30-40% censoring, N\_{rep}=30.
* **Scenarios (4):**
  1. Linear f(\mathbf{x}), Uncorrelated X.
  2. Linear f(\mathbf{x}), Correlated X.
  3. Non-Linear f(\mathbf{x}), Uncorrelated X.
  4. Non-Linear f(\mathbf{x}), Correlated X.
* **Expected Outcome:** Relevant variables (X1, X2, X3) should consistently rank higher than noise variables (X4, X5).

**Scenarios:**

1. **Linear Uncorrelated:**
   * f(x)=1⋅X1​+1⋅X2​−1⋅X3​+0⋅X4​+0⋅X5​
   * Features X1​,…,X5​∼N(0,1), all independent.
   * *Expected Ranking:* X1, X2, X3 > X4, X5
2. **Linear Correlated:**
   * f(x) same as Scenario 1.
   * Features: Generate X1​,…,X5​ from N5​(0,Σ) where Σ has some correlation, e.g., block correlation where (X1, X2) are correlated, (X3, X4) are correlated, X5 independent.
   * *Expected Ranking:* Still X1, X2, X3 > X4, X5, but correlations might slightly affect ghost estimation and relevance scores.
3. **Non-Linear Uncorrelated:**
   * f(x)=1⋅X1​+sin(π⋅X2​)+0.5⋅X32​+0⋅X4​+0⋅X5​ (Example non-linearity)
   * Features X1​,…,X5​∼N(0,1), all independent.
   * *Expected Ranking:* X1, X2, X3 > X4, X5 (assuming the non-linear terms are strong enough).
4. **Non-Linear Correlated:**
   * f(x) same as Scenario 3.
   * Features: Correlated as in Scenario 2.
   * *Expected Ranking:* X1, X2, X3 > X4, X5.

**7. Current Status (End of Day 3 / Start of Day 4)**

* Python environment is fixed.
* Core classes (SurvivalDataGenerator, NonLinearSurvivalModel, GhostVariableEstimator, GhostCoxInterpreter, ExperimentRunner) are implemented and tested end-to-end.
* Normalization using Var(\hat{f}\_{\text{test}}) is implemented.
* Simulation Study 1 scenarios are defined.
* **Action:** Currently running Simulation Study 1 (runner.run\_experiment()).

**8. Agreed 2-Week Plan Outline**

* **Week 1 (Days 1-7):** Environment fix, Methodology refinement, Code implementation & testing, Design & Run Simulation 1, Analyze Sim 1 results (ranking), Prep & Run Real Data Example, Draft Intro/Methodology/Sim/Real Data sections.
* **Week 2 (Days 8-14):** Draft Literature Review & Discussion, Draft Conclusion & Abstract, Consolidate & Format thesis, Code cleanup, Advisor feedback & Revisions, Final Polish & Submission.

**Okay, I understand. While I can't directly create or edit a Google Doc for you, I can provide the detailed 2-week plan we just discussed in a structured format right here. You can easily copy and paste this into a Google Doc, text editor, or project management tool.**

**Here is the proposed 2-week plan:**

**GhostCox Thesis - Aggressive 2-Week Plan**

**Goal: Produce a complete thesis draft demonstrating the GhostCox method, including implementation, basic simulation results, one real-data application, and core written chapters within 14 days.**

**Prerequisite: Resolve the Python environment error in ghost\_cox.ipynb *immediately* (run uninstall/reinstall cell, restart kernel).**

**Week 1: Core Implementation, Basic Validation & Initial Writing (Days 1-7)**

* **Day 1: Environment Fix & Methodology Refinement** 
  + **✅ Task 1: Fix Python environment (pip install in activated venv, restart kernel, verify imports).**
  + **Task 2: Refine Methodology section (2.3/2.4) in thesis\_draft.pdf. Define GhostCox stages, initial normalization choice (e.g., Var(f^​(Xtest​))), default ghost estimator (e.g., RF).**
* **Day 2: Code Solidification** 
  + **Task 1: Implement chosen NormalizationFactor in GhostCoxInterpreter.**
  + **Task 2: Test the full code pipeline (DataGen -> SurvivalModel -> GhostEstimator -> Interpreter) on simple simulated data. Ensure relevance scores are generated.**
* **Day 3: Simulation Study 1 Design & Setup** 
  + **Task 1: Design focused Simulation Study 1 (verify ranking):** 
    - **Scenario 1: Linear f(x), uncorrelated X.**
    - **Scenario 2: Linear f(x), correlated X.**
    - **Scenario 3: Simple Non-linear f(x), uncorrelated X.**
    - **Scenario 4: Simple Non-linear f(x), correlated X.**
    - ***(Keep N, P, censoring fixed initially)*.**
  + **Task 2: Set up scenarios in ExperimentRunner.**
* **Day 4: Run Simulation 1 & Real Data Prep** 
  + **Task 1: Start running Simulation Study 1 (n\_replicates=20-30).**
  + **Task 2: Identify & preprocess one real dataset (e.g., whas500, gbsg2).**
* **Day 5: Analyze Simulation 1 & Draft Results** 
  + **Task 1: Process Simulation 1 results. Focus on variable ranking accuracy. Create basic plots (avg. ranks).**
  + **Task 2: Draft "Simulation Study" section (setup & results).**
* **Day 6: Real Data Application** 
  + **Task 1: Split real data (train/test).**
  + **Task 2: Fit NonLinearSurvivalModel (e.g., RSF) on train set.**
  + **Task 3: Run GhostCoxInterpreter (e.g., RF ghosts) on test set.**
  + **Task 4: Fit standard CoxPH on train set for comparison.**
* **Day 7: Draft Real Data Results & Refine Chapters** 
  + **Task 1: Draft "Real Data Application" section (data description, model fits, GhostCox ranking, qualitative comparison to CoxPH).**
  + **Task 2: Review & refine Introduction and Methodology chapters.**

**Week 2: Extended Evaluation & Writing (Days 8-14)**

* **Day 8: Literature & Discussion Outline** 
  + **Task 1: Draft Literature Review (position GhostCox relative to Cox, robustness papers, SHAP, Ghost Variables ).**
  + **Task 2: Outline Discussion (interpret results, advantages, limitations, comparison to SHAP-theory).**
* **Day 9: Write Discussion** 
  + **Task 1: Write full draft of Discussion section.**
* **Day 10: Write Conclusion & Abstract** 
  + **Task 1: Write Conclusion (summary, contributions, limitations, future work).**
  + **Task 2: Write Abstract.**
* **Day 11: Consolidate & Format** 
  + **Task 1: Combine all sections. Check flow, consistency.**
  + **Task 2: Format references, tables, figures (ensure LaTeX for math). Add captions.**
* **Day 12: Code Cleanup & Document Review** 
  + **Task 1: Clean up ghost\_cox.ipynb, add comments, ensure outputs match thesis.**
  + **Task 2: Read through entire draft for clarity, grammar, typos. Check claims vs. results/literature.**
* **Day 13: Advisor Feedback & Revisions** 
  + **Task 1: Urgent: Share complete draft & code with advisor for feedback.**
  + **Task 2: Incorporate feedback and revise.**
* **Day 14: Final Polish & Submission** 
  + **Task 1: Final proofread, check university formatting rules.**
  + **Task 2: Generate final PDF, submit thesis & code.**

**(Next Steps: Monitor simulation run, prepare real dataset, analyze simulation results - Day 4/5 tasks)**