CryptoCredit: Securely Training Fair Models

Leo de Castro  
MIT CSAIL  
Cambridge, Massachusetts  
ldec@mit.edu

Jiahao Chen  
J.P.Morgan AI Research  
New York, New York  
jiahao.chen@jpmorgan.com

Antigoni Polychroniadou  
J.P.Morgan AI Research  
New York, New York  
antigoni.poly@jpmorgan.com

ABSTRACT
When developing models for regulated decision making, sensitive features like age, race and gender cannot be used and must be obscured from model developers to prevent bias. However, the remaining features still need to be tested for correlation with sensitive features, which can only be done with the knowledge of those features. We resolve this dilemma using a fully homomorphic encryption scheme, allowing model developers to train linear regression and logistic regression models and test them for possible bias without ever revealing the sensitive features in the clear. We demonstrate how it can be applied to leave-one-out regression testing, and show using the adult income data set that our method is practical to run.

1 INTRODUCTION
Major institutions are incorporating machine learning systems to drive decisions more efficiently and accurately in tasks such as loan approval, prediction of defaults and fraud detection. While machine learning algorithms progress rapidly and transform the industry, they introduce the following challenge:

How can we ensure that machine learning algorithms do not inadvertently perpetuate bias that is inherent in the training data?

In a well-known example of historical bias appearing in a machine learning model, in 2016 a model to predict future criminal recidivism turned out to be biased against African-Americans [2, 16].

More and more financial institutions have developed teams of machine learning experts to develop models of creditworthiness. These models are trained on historical credit data, which raises concerns that these models could perpetuate historical biases against marginalized groups [4, 15, 22]. To address this concern, some consumer financial institutions have also developed compliance teams that are responsible for ensuring that the methods to determine credit do not violate federal anti-discrimination laws.

One of the basic distinctions between the compliance team and the machine learning team is that the compliance team is able to see all fields in the customer database, including those corresponding to classes protected by anti-discrimination law (i.e., age, race, sex, etc.) [9], while the machine learning team is only able to see a subset of the data fields. The compliance team determines which features the machine learning team is allowed to see and, subsequently, use to train their models.

Obviously, the fields directly corresponding to the protected classes are not allowed to be viewed by the machine learning team. However, biases in the data could allow a model to implicitly learn these protected fields. This is represented by previously unprotected fields having strong correlation with the protected field. A classic example of this phenomenon is race and zip code, known as redlining [12, 19]. Even if a machine learning model does not directly use the race field during training, if the zip code field is also used during training, the model could implicitly learn the race field since, in many areas, race and zip codes are highly correlated. This would be a reason that the compliance team would remove the zip code field from the fields available to the machine learning team.

However, discovering these correlations is often difficult, since the bulk of the machine learning expertise is concentrated in the machine learning teams, who are not allowed to see the protected fields. The machine learning team could potentially assess whether or not there is unconscious bias in the training data by building models to predict the prohibitive features from the rest of the features. This state of affairs introduces another challenge:

How can we ensure that privacy of prohibitive features is preserved when the model is trained?

To solve this seemingly paradoxical problem, we use cryptography. Modern cryptography employs a rigorous mathematical approach to compute on encrypted data. A cryptographic tool, called Fully Homomorphic Encryption (FHE) solves the problem of secure computation between two parties, Alice and Bob. More specifically, Alice has an input $x$ and Bob has a function $f$. Alice wishes to outsource the computation of $f(x)$ to Bob without revealing $x$. FHE allows Alice to encrypt her input $x$ into a ciphertext $ct_x$ that
can be safely sent to Bob. Bob can then homomorphically evaluate
the function \( f \) on the ciphertext \( c_t \), to produce a new ciphertext
\( c_{f(x)} \) that encrypts \( f(x) \). This resulting ciphertext \( c_{f(x)} \) is re-
turned to Alice, who decrypts to obtain \( f(x) \).

### 1.1 Our Contributions

We frame the problem of securely developing a fair ML model as
a secure computation outsourcing problem. More specifically, the
machine learning team has training data \( X \) that must be kept private.
At the same time, the compliance team has limited computing re-
sources and needs to outsource the training of the ML model to
the machine learning team. This problem can be solved with FHE,
since the compliance team can encrypt the training data set \( X \) and
send only the ciphertext to the machine learning team. The ma-
chine learning team then trains their model over the encrypted
data set, then returns the model to the compliance team.

Not all aspects of the training data set are hidden from the ma-
chine learning team. The feature labels (or schema) are revealed to
the machine learning team, and, in addition, the feature labels are
indicated as either protected or unprotected. Only the unprotected
features are used in training the ML models, but the protected fea-
tures are very useful in evaluating fairness tests on these ML mod-
els.

In Section 3, we introduce one of the simplest methods for fair-
ness testing in regression models, which are commonly used in
econometrics and other financial applications [3, 31]. The basic
idea is leave-one-out regression, in which we train two regression
models using the same feature set \( X \), but including also \( S \) in only
one of the models. The idea is then to test whether or not the coeffi-
cients in these two models show statistically significant differences.
We present the classical approach to hypothesis testing using Wald
tests to test the equality of regression coefficients [37]. In Section 4,
we show how to use our privacy preserving techniques to perform
these LOO regression by training on the encrypted features. In
Section 5, we apply our analysis of leave-one-out regression and
Wald’s test to the UCI Adult data set [17, 25], a well-studied data set
of extracted census data with many fields similar to a loan applica-
tion. When training in a privacy preserving manner a linear model
on this data set, we found that our test revealed biases in the train-
ing data that were learned by the model. Specifically, we found the
linear model learned the correlation between the “Age” feature and
the “Marital Status” feature, with the “Never Married” feature and
the “Widowed” feature having the strongest correlation. This aligns
nicely with intuition.

In addition, we found that certain models were better than oth-
ers at detecting biases in the training data. Specifically, while the
linear regression model was able to detect strong biases in the train-
ing data, the logistic regression model did not seem to learn this
correlation, despite having essentially the same accuracy as the
linear model.

### 1.2 Related Work

*Fairness in artificial intelligence.* In recent years, concerns over
fairness in automated decision-making systems has received well-
deserved attention [29, 39], particularly in predictive policing [2, 6,
11, 16, 28] and credit decisioning [1, 10, 15]. Fairness is enshrined in
regulatory requirements for some applications, notably in credit [9,
15] and hiring decisions [13, 19, 26, 32]. Several authors have criti-
cized the use of automated decision making without regard to fair-
ness or interpretability needs in other applications [14, 35, 36, 40].
However, most of the work focuses on binary outcome variables,
and less so on continuous outcomes which result from regression
studies. Nevertheless, the work of Kleinberg et al. [24], Pleiss et al.
[30] on calibration notions of fairness deserve special attention, as
being applicable specifically to regression models with continuous
outcomes.

As described in the introduction, our work is based on a direct
analysis of the coefficients of regression models, as is common in
econometric analysis with a history going back at least to Conway
and Roberts [13]. A similar (but not identical) treatment of linear
regression models is given in Pope and Sydnor [31]. Similar ideas
also appear in the leave one covariate out (LOCO) method of Lei
et al. [27] for explaining variable-level importance.

### 2 BACKGROUND

#### 2.1 Notation & Terminology

**Definition 2.1 (Data Schema).** For a data set \( D \), denote the
schema for \( D \) as \( S_D \). The schema of the dataset consists of the labels
of the features of the dataset.

**Definition 2.2 (Data Set Select Operation).** Let \( D \) be a data
set with schema \( S_D \), and let \( S_T \subseteq S_D \) denote a subset of the features
in \( S_D \). The Select operation takes in a subset of features \( S_T \) along
with a data set \( D \) and produces a new dataset \( T \) with the same entries
as in \( D \) except with only the features in \( S_T \).

\[
T \leftarrow \text{Select}(S_T, D)
\]

For a subset \( T \) of a set \( S \), denote the compliment of \( T \) in \( S \) by \( T^c \),
where \( T^c \cap T = \emptyset \) and \( T^c \cup T = S \). We also denote \( T^c = S \setminus T \).

#### 2.2 Leave-One-Out Regression

In this section, we describe leave-one-out (LOO) regression, which
is fundamental tool for the statistical tests we perform on machine
learning models in this work. The goal of LOO regression is to
determine how a given feature \( f \) in a data set \( D \) is related to the
other features in the data set. The high-level idea is to train a model
\( M \) on the data set \( D \), then train a second model \( M' \) on the data
set \( D \setminus \{f\} \), then measure how the coefficients of the model \( M' \)
differ from the original model \( M \). If the weights corresponding to
another feature \( f' \) change significantly, then we conclude that \( f' \)
and \( f \) are correlated.

#### 2.3 Cryptographic Notions

In this section, we give high level definitions for the cryptographic
primitives that we will use in the protocols below. All of these primiti-
ves will be used in a black-box manner.

A fully homomorphic encryption scheme allows us to perform
arbitrary computations on encrypted data without decrypting it.
Although the notion was introduced back in [33], the first imple-
mentation is due to a much later breakthrough of [20], and has now
been followed with much exciting activity [7, 8, 38]. In a nutshell,
the data owner can encrypt a message \( m \) using a private key to pro-
duce some ciphertext \( c \), which can be sent to some external party.
The external party can then compute any function \( f \) on the ciphertext, which will result in a new ciphertext \( c' \) that corresponds to \( f(m) \). The data owner can then decrypt \( c' \) to obtain \( f(m) \).

FHE solves the problem of secure computation between two parties, a compliance team Comp and a machine learning team ML: Comp encrypts their input and sends it to ML, who can homomorphically evaluate the function on Comp’s encrypted input and its own input, sending only the resulting ciphertext back to Comp, who can decrypt.

We formalize the algorithms that comprise a fully homomorphic encryption scheme.

**Definition 2.3 (Fully Homomorphic Encryption).** A fully homomorphic encryption scheme

\[
\text{FHE} = (\text{KeyGen}, \text{Encrypt}, \text{Eval}, \text{Decrypt})
\]

is a set of probabilistic polynomial-time algorithms defined as follows:

- \( \text{KeyGen}(1^\lambda) \rightarrow (sk, pk, evk) \)
  
  Given the security parameter \( \lambda \), outputs a key pair consisting of a public encryption key \( pk \), a secret decryption key \( sk \), and an evaluation key \( evk \).

- \( \text{Encrypt}(pk, m) \rightarrow ct \)
  
  Given a message \( m \in \mathbb{M} \) and an encryption key \( pk \), outputs a ciphertext \( ct \).

- \( \text{Eval}(evk, f, ct_1, ct_2, \ldots, ct_n) \rightarrow ct' \)
  
  Given the evaluation key, a description of a function \( f: \mathbb{M}^n \rightarrow \mathbb{M} \), and \( n \) ciphertexts encrypting messages \( m_1, \ldots, m_n \), outputs the result ciphertext \( ct' \) encrypting \( m' = f(m_1, \ldots, m_n) \).

- \( \text{Decrypt}(sk, ct) = m \)
  
  Given the secret decryption key and a ciphertext \( ct \) encrypting \( m \), outputs \( m \).

We now briefly provide intuition for the security definition of an FHE scheme. Note that our only concern is security for the compliance team, and we will not consider any security notion for the machine learning team. The standard and most basic security definition of an encryption scheme is known as semantic security.

**Definition 2.4 (Semantic Security [21]).** An FHE scheme satisfies semantic security if no probabilistic polynomial time adversary \( A \) can succeed in the following task with probability greater than \( 1/2 + \negl(\lambda) \), where \( \negl(\lambda) \) is a function that shrinks faster than \( 1/poly(\lambda) \) for every polynomial poly.

\( A \) takes as input \( pk \) and \( evk \) and produces two messages \( m_1, m_2 \in \mathbb{M} \). \( A \) then takes in the encryptions of \( m_1 \) and \( m_2 \), denoted \( ct_1 \) and \( ct_2 \), in a random order. \( A \) must then decide which ciphertext is an encryption of \( m_1 \) and which ciphertext is an encryption of \( m_2 \).

Intuitively, Definition 2.4 says that the encryptions of two messages should be indistinguishable, even for adversarially chosen messages. As encryption schemes, all known FHE schemes satisfy this security definition.

---

1This is often referred to as circuit privacy or function privacy in the FHE literature.

### 3 LEAVE-ONE-OUT REGRESSION AND WALD’S TEST

In this section, we present a series of tests that we will perform on trained models to approach a model that is independent of a set of protected features. The setting is formally similar to that of covariate adjustment [34], and is also similar, but not identical to, the treatment of Pope and Sydnor [31]. Let \( S = \{s_1, s_2, \ldots, s_n\} \) be set of sensitive features, \( X = (x_1, x_2, \ldots, x_n) \) be the set of ordinary features, and \( Y \) be the target variable (with prediction \( \hat{Y} \)).

The idea is to create two linear regression models

\[
M : Y = X\beta + Sy + e_M, \quad \text{and} \quad M' : Y = X\xi + e_M',
\]

where \( \beta \in \mathbb{R}^n, Y \in \mathbb{R}^n, \) and \( \xi \in \mathbb{R}^n \) are the vectors of regression coefficients and \( e_M, e_M' \) are the corresponding noise terms.

Alternatively, we may also consider a similar pair of logistic regression models

\[
M : \log \frac{1 - \hat{Y}^L}{\hat{Y}^L} = X\beta + Sy + e_M, \quad \text{and} \quad M' : \log \frac{1 - \hat{Y}^L}{\hat{Y}^L} = X\xi + e_M',
\]

with similarly defined symbols. To abuse notation, we distinguish linear regression from logistic regression modeling by discussion context, keeping the same notation for consistency of discussion.

We can now use the generalized Wald’s test to test constraints on the regression parameters [37]. Let \( N \) be number of data points, \( n_X \) be the number of ordinary features, which is the number of columns of \( X, n_S \) be the number of sensitive features, which is the number of columns of \( S, D \) be a diagonal matrix with the first \( n_X + n_S \) entries equal to \( 1/(N-n_X-n_S) \) and the next \( n_X \) entries equal to \( N-n_X \), whose ith diagonal entry counts the degrees of freedom used to estimate the ith coefficient, and \( \theta = (\beta^T \, \gamma^T \, \xi^T)^T \in \mathbb{R}^{2n_X+n_S} \) be the vector of all coefficients belonging to both models. Furthermore, let \( \hat{\theta} = (\hat{\beta}^T \, \hat{\gamma}^T \, \hat{\xi}^T)^T \in \mathbb{R}^{2n_X+n_S} \) be the estimators for the parameters, and \( \mathbb{E}(\cdot) \) denote the expectation value. To test hypotheses of the form \( H\theta = c \), where \( H \) is a matrix of size \( k \times (2n_X + n_S) \), construct the test statistic

\[
W = (H\hat{\theta} - c)^T (HD^{-1/2}\hat{\Sigma}D^{1/2}H^T)^{-1} (H\hat{\theta} - c) \sim \chi^2_k,
\]

where \( \hat{\theta} \) are the estimates for the parameters, and \( \hat{\Sigma} = \mathbb{E}((\hat{\theta} - \mathbb{E}\hat{\theta})(\hat{\theta} - \mathbb{E}\hat{\theta})^T) \) is the estimated covariance matrix for the coefficients.

This general form of Wald’s test allows us to test multiple equalities simultaneously. The multiple equality testing can be useful for testing hypotheses like \( \beta = \xi \), testing equalities of all the coefficients simultaneously, \( \gamma = 0 \) for testing for the null effect of \( S \), or more fine-grained tests where multiple dummy variables encoding the same categorical variable can be tested. Nevertheless, in this paper, we focus exclusively on hypothesis involving one
equality test for simplicity, while being illustrative of the general case. More specifically, we test pairwise equality hypotheses of the form $β_i = ζ_i$, each to be tested against their corresponding alternative hypothesis $β_i ≠ ζ_i$. Let $H_i$ be the $1 \times (2X + n_S)$ matrix with the $(1, i)$th entry equal to +1 and $(1, n_S + n_i + 1)$th entry equal to -1, and all other entries being 0. The corresponding Wald statistic is then
\[
W = \frac{\sum_β β_i^2}{N-n_X-n_S} - \frac{\sum_β β_iζ_i + ζ_iβ_i}{\sqrt{(N-n_X-n_S)(N-n_X)}} + \frac{\sum_ζ ζ_i^2}{N-n_X} \tag{6}
\]
where we approximate $Σ_β β_iζ_i = E((β_i - Eβ_i)(ζ_i - Eζ_i)) ≈ 0$, and note that $se(β_i) = \sqrt{Σ_β β_i(1-n_X-n_S)}$ and $se(ζ_i) = \sqrt{Σ_ζ ζ_i(1-n_X)}$ are exactly the definitions of the standard errors for the respective coefficients.

From (6), we have Algorithm 1, which allows us to determine the coefficients in a model that are correlated with a protected feature.

Algorithm 1 Univariate Wald Test, denoted WaldTest

Input: Two models $M$ and $M’$, and significance level $0 < α < 1$.

1. The models $M$ and $M’$ must satisfy the constraint that the training data $D$ for $M$ and training data $D’$ for model $M’$ satisfy the relation that $S_{D’} = S_D \cup \{p\}$, where $p$ is the protected feature.
2. Compute the critical value $W^*$ from the $χ_1^2$-distribution. For the standard $α = 0.05$, $W^* = 3.9$.
3. $S ← ∅$
4. for each feature $i ∈ S_D$ do
5. Estimate the coefficients $β_i$ and $ζ_i$ for the $i$th feature in models $M$ and $M’$ respectively.
6. Compute the standard errors $se(β_i)$ and $se(ζ_i)$ of the coefficients $β_i$ and $ζ_i$.
7. Compute the Wald statistic, $W_i = \frac{(β_i - ζ_i)^2}{se(β_i)^2 + se(ζ_i)^2}$, of (6), testing the null hypothesis $β_i = ζ_i$ against its alternative $β_i ≠ ζ_i$.
8. if $W_i > W^*$ then
9. $S ← S \cup \{i\}$
10. end if
11. end for
Output: $S$

4 A SECURE FAIRNESS SYSTEM

In this section, we present the design of our secure machine learning fairness protocol. The protocol is run between two parties: the machine learning party $ML$ and the compliance party $Comp$. $Comp$ begins with a dataset $D = P \cup U$, where each entry in $D$ consists of a set of sensitive features $P$ and a set of unsensitive features $U$. We call features that are a priori prohibited from use "sensitive features", and features that are a priori permitted to use "unsensitive features". After the fairness tests are performed, we will have features in the unsensitive set that are correlated with the features in the sensitive set. None of these features may be used in the training data of the final model, and we call these features "protected features". All features that are not protected are "unprotected features."

The sets $P$ and $U$ are denoted by schema $S_P$ and $S_U$. At a high level, the goal of this protocol is to produce a model $M$ that is independent of all features in $P$. The approach is to identify features in $U$ that encode features in $P$. The protocol proceeds in a series of rounds, and at the end of each round the set $U$ is updated to exclude features correlated with features $P$. When $U$ is not updated at the end of the round, the model trained on the features in the most recent $U$ is accepted as a fair model.

The high level security goal of the protocol is to hide the values of the protected features from the machine learning team. By hiding these values, we can argue that these values could not have implicitly influenced their decision to include or exclude unprotected features.

4.1 A Single Round of the Fairness Protocol

A single round of the protocol is given in Algorithms 2 and 3. The full protocol, presented in Algorithms 4 and 5 in Section 4.2, primarily consists of running Algorithms 2 and 3 repeatedly until the resulting model passes the Wald test.

Beginning with Algorithm 2, the inputs to a single round for the compliance team is a data set $D$, a schema of sensitive features $S_P$, a schema of unprotected features $S_U$ and an FHE key pair $pk$ and $sk$. The schema of sensitive and unprotected features must partition the data set $D$. In other words, the two schema must satisfy $0 = S_P \cap S_U$ and $S_D = S_P \cup S_U$, where $S_D$ is the schema for the whole data set. In each round of the protocol, the compliance team sends the encrypted dataset to the machine learning team along with the partition of sensitive and unprotected features, represented by $S_P$ and $S_U$. The machine learning team then trains a model on the unprotected features using a fully homomorphic encryption scheme. The result of this training is an encrypted model that the machine learning team returns to the compliance team. Since the compliance team has the FHE secret key $sk$, they can decrypt the encrypted model to obtain the model weights in the clear.

In addition to the model $M$ trained on the unprotected features, the compliance team also receives a series of models $M_i^f$ for each sensitive feature $p_i ∈ S_P$. The model $M_i^f$ is trained on the dataset $D_i^f = Select(S_U \cup \{p_i\}, D)$, which is all unprotected features plus the single sensitive feature $p_i$. For each of these models $M_i^f$, the compliance team performs the univariate Wald test as described in Algorithm 1. If there are features in $S_U$ that are correlated with the sensitive feature $p_i$ and the model $M$ has learned this correlation, the Wald test will detect this by examining the difference in the weights $M$ from the weights of $M_i^f$. The output of this test will be the set of features $S_i^f ⊆ S_U$ that are correlated with $p_i$. The features in $S_i^f$ are now considered protected, and in subsequent
Algorithm 2 Single Round of Fairness Protocol for Comp, denoted CompOneRound

**Input:** Data set \( D \)
- Schema of sensitive features \( S_P \)
- Schema of unprotected features \( S_U \)
- FHE public key \( \text{pk} \)
- FHE secret key \( sk \)

1. Encrypt the dataset \( \text{ct}\_D \leftarrow \text{Encrypt} (\text{pk}, D) \).
2. Send the encrypted dataset \( \text{ct}\_D \) along with the two schemas \( S_P \) and \( S_U \) to ML.
3. Receive encrypted model \( \text{ct}\_M \) trained on \( S_U \) from ML.
4. \( M \leftarrow \text{FHE.Decrypt} (sk, \text{ct}\_M) \).
5. \( \hat{S} \leftarrow \emptyset \).
6. for \( p_i \in S_P \) do
7. \( M_{p_i} \leftarrow \text{FHE.Decrypt} (sk, \text{ct}\_M) \).
8. \( \hat{S}_i \leftarrow \text{WaldTest} (M, M_{p_i}) \).
9. \( S_i \leftarrow \hat{S}_i \cup \hat{S} \).
10. end for

**Output:** \( M, \hat{S} \)

### 4.2 Full Fairness Protocol

Building off of the algorithms in Section 4.1, we can define our full fairness protocol in Algorithms 4 and 5. These algorithms follow naturally from the discussion in Section 4.1, as they are essentially just running the single round algorithms until the machine learning team produces a model that is sufficiently independent of the sensitive features.

At the end of each round, the compliance team obtains a candidate model and a set of protected features \( \hat{S} \) in the candidate model’s training data that are correlated with the sensitive features. In the next round, the compliance team removes the features in \( \hat{S} \) from the unprotected features \( S_U \). This operation must be performed homomorphically over the encrypted data set \( \text{ct}\_D \).

Algorithm 3 Single Round of the Fairness Protocol for ML, denoted MLOneRound

**Input:** FHE evaluation key \( \text{evk} \)
-learning algorithm \( \text{Learn} \).
1. Receive the encrypted dataset \( \text{ct}\_D \).
2. Receive the schema for the sensitive and unprotected features \( S_P \) and \( S_U \).
3. Select the features for the training data \( S_T \subseteq S_U \).
4. \( \text{ct}\_M \leftarrow \text{Eval}(\text{evk}, \text{Select}(S_U, \cdot), \text{ct}\_D) \).
5. \( \text{ct}\_M \leftarrow \text{Eval}(\text{evk}, \text{Learn}(\cdot), \text{ct}\_M) \).
6. Send \( \text{ct}\_M \) to Comp.
7. for \( p_i \in S_P \) do
8. \( S_{\hat{U}_i} \leftarrow S_U \cup \{p_i\} \).
9. \( \text{ct}\_U_i \leftarrow \text{Eval}(\text{evk}, \text{Select}(S_{\hat{U}_i}, \cdot), \text{ct}\_D) \).
10. \( \text{ct}\_M_{p_i} \leftarrow \text{Eval}(\text{Learn}(\cdot), \text{ct}\_U_i) \).
11. Send \( \text{ct}\_M_{p_i} \) to Comp.
12. end for

Algorithm 4 Full Fairness Protocol for Comp

**Input:** Data set \( D \)
- Schema of protected features \( S_P \)
- Security parameter \( \lambda \)
1. \( (sk, pk, evk) \leftarrow \text{FHE.KeyGen}(1^\lambda) \).
2. Send \( evk \) to ML.
3. while True do
4. Let \( S_T \leftarrow S_P \).
5. \( M, \hat{S} \leftarrow \text{CompOneRound}(D, S_P, S_U, pk, sk) \).
6. if \( \hat{S} \) equals \( \emptyset \) then
7. Send Terminate signal to ML.
8. Return: \( M \).
9. end if
10. Send Continue signal to ML.
11. \( S_U \leftarrow S_U \setminus \hat{S} \).
12. end while

We can see that Algorithm 4 always terminates, since \( \hat{S} \subseteq S_U \), so \( S_U \) must shrink with each iteration of the while loop. If \( S_U \) does not shrink, then it must be that \( \hat{S} = \emptyset \), so the while loop terminates. Similarly, we have that Algorithm 5 terminates exactly when Algorithm 4 terminates. Since we have already argued that Algorithm 4 terminates, we have that Algorithm 5 terminates as well.
We began by training two models using the linear regression learning algorithm. The first model $M$ was trained on $\mathcal{U}$ and achieves an accuracy of $83.5\%$. The second model $M'$ was trained on the dataset $\mathcal{U} \cup \{\text{Age}\}$ and also achieved an accuracy of $83\%$. The train-test split of the data entries were identical for both models.

### 5 CASE STUDY: UCI ADULT DATASET

In this section, we implement the system described in Section 4 to train a linear model on the adult income dataset, a popular data set which is available at the machine learning dataset repository of the University of California at Irvine [17, 25]. The UCI Adult data set contains 14 attributes extracted from 1994 census data, of which 3 are federally protected classes: age, sex, and race. For simplicity, we consider age only in our study. For the remaining 11 features, we use a standard dummy variable transformation of the categorical variables, also known as one-hot encoding, encoding a categorical variable with $\text{m}$ levels into $\text{m}−1$ binary variables, dropping one of the dummy variables to avoid collinearity issues in the subsequent regression. This transformation of the categorical variables yielded a total of $n_X = 87$ features, being a mix of numeric and binary variables. Dropping all rows with missing entries yields a final data set of 30,162 rows in the training subset and 15,060 rows in the testing subset, which we use in our analysis. The target feature is a binary variable indicating if the person corresponding to the entry has an income of at least $\$50,000 per year. Denote this set of $n_X = 87$ features by Adult and the age protected class as $\text{Age}_1$, represented by a single positive integer. We begin our experiments with $\mathcal{U} = \text{Adult}$ and $\mathcal{P} = \{\text{Age}\}$.

#### 5.1 Linear Regression Model

We began by training two models using the linear regression learning algorithm. The first model $M$ was trained on $\mathcal{U}$ and achieves an accuracy of $83.5\%$. The second model $M'$ was trained on the dataset $\mathcal{U} \cup \{\text{Age}\}$ and also achieved an accuracy of $83\%$. The train-test split of the data entries were identical for both models.

We then compute the Wald test on inputs $\mathcal{M}$ and $\mathcal{M}'$ to obtain the set of features that the model $M$ correlates with Age, and calculate the corresponding $p$-values to test at the usual $\alpha = 0.05$ level of significance. We found two features that changed significantly out of the $n_X = 87$ coefficients, and these results are shown in full in Table 2, with the significant features summarized in Table 1. These features are the Never-married and Widowed categories of marital status. The significance of the Never-married category persists even after applying the very conservative Bonferroni correction for multiple hypothesis testing [18], at the $\alpha/n_X = 0.0006$ level.

To account for this correlation in the marital status feature, we reduce the marital status feature from seven categories to a single binary feature of [Unmarried, Married] in the next round. We repeat this experiment with the unprotected features updated to reflect this change to the marital status. Both of the resulting models remained around $83\%$ accuracy and none of the features between the two models changed significantly. We can then conclude that this final model trained on the unprotected Adult data set with the updated marital status feature is independent of the protected Age feature.

#### 5.2 Logistic Regression Model

While biases may be present in the Adult data set, not all models will learn this bias equally. To illustrate this, we performed the leave-one-out regression experiment with $\mathcal{U} = \text{Adult}$ and $\mathcal{P} = \{\text{Age}\}$ using a logistic regression learning model. Once again, the two models achieved $83\%$ accuracy, but, unlike the linear regression model, the weights in the logistic regression model did not change significantly when the Age feature was added. Based on this test, we can conclude that this logistic regression model has not detected the biases in the Adult data set that the linear regression model was able to learn. It is an interesting question for future work to consider how different models learn biases. The full results of this experiment are shown in Table 3.

#### 5.3 Encrypted Implementation

The protocol in Section 4 is agnostic to the underlying homomorphic encryption scheme as well as the method in which the learning algorithm is implemented using the chosen homomorphic encryption scheme. Implementations of logistic regression using homomorphic encryption abound [5, 23]. We implemented secure logistic regression training on the Adult data set using the HELR library from [23] on a standard machine. While the full running time for training a single logistic regression model was about 45

| Feature          | Original Coefficient | Coefficient with Age | $W$   | $p$     |
|------------------|----------------------|----------------------|-------|--------|
| Never Married    | -0.0775              | -0.0063              | 12.041| 0.0005 |
| Widowed          | 0.0895               | 0.009                | 4.445 | 0.0350 |

Table 1: Features with the largest difference in standard errors between the OLS model without Age and the OLS model with Age. These were the only features that changed more than two standard errors.

---

[^1]: The trivial accuracy for the UCI Adult data set is 75.4%, which is the proportion of persons in the data set making less than $\$50,000 dollars per year.

[^2]: We used an AWS instance with a 3.0 GHz CPU with 8 cores and 32 GB of RAM.
minutes, only about three minutes of computation was actually spent progressing the logistic regression function. The remainder of the computation time was spent on the bootstrapping operation. This operation does not advance the underlying computation; rather, it is a technical operation necessary to maintain the correctness of the result when the function circuit is sufficiently deep. However, our goal in this setting is not to hide the learning algorithm from the compliance team, only to minimize the amount of machine learning expertise on the compliance side. This allows us to employ much more efficient interactive bootstrapping routine that can be run by the compliance team. By introducing interaction, we can effectively eliminate almost all of the computation time spent on the bootstrapping operation. It is an interesting direction for future work to optimize this fairness outsourcing protocol.

Disclaimer. This paper was prepared for informational purposes by the Artificial Intelligence Research group of JPMorgan Chase & Co and its affiliates (“JP Morgan”), and is not a product of the Research Department of JP Morgan. JP Morgan makes no representation, warranty or guaranty as to the accuracy or completeness of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction, and shall not constitute a solicitation under any jurisdiction or to any person, if such solicitation under such jurisdiction or to such person would be unlawful.

REFERENCES

[1] Grace M Ababamad. 2019. The Fallacy of Equating “Blindness” with Fairness: Ensuring Trust in Machine Learning Applications to Consumer Credit. MS. MIT.
[2] Julia Avgwin, Jeff Lurya, Saturna Matha, and Kittiruch Charan. 2016. Machine bias ProPublica.
[3] Arthur P. Baines and Marsha J. Courchane. 2014. Fair Lending: Implications for the Indirect Auto Finance Market. Charles River Associates.
[4] Solon Barocas and Andrew Selbst. 2016. Big Data’s Disparate Impact. California Law Rev. 104, 1 (2016), 671–729.
[5] Flavio Bergamaschi, Shai Halevi, Tzipora T. Halevi, and Hamish Hunt. 2019. Homomorphic Training of 30,000 Logistic Regression Models. In Applied Cryptography and Network Security, Robert H. Deng, Valérie Gauthier-Umaña, Martin Ochoa, and Moi Yung (Eds.). Springer International Publishing, Cham, 592–611.
[6] Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. 2018. Fairness in Criminal Justice Risk Assessments: The State of the Art. Sociological Methods & Research (2018), in press.
[7] Žekra Brakerski and Vinod Vaikuntanathan. 2014. (Leveled) Fully Homomorphic Encryption without Bootstrapping. ACM Trans. Comput. Theory 6, 3, Article 13 (July 2014), 36 pages.
[8] Žekra Brakerski and Vinod Vaikuntanathan. 2011. Efficient Fully Homomorphic Encryption from (Standard) LWE. In IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS. Rafael Ostrovsky (Ed.). IEEE Computer Society, Palm Springs, CA, 97–106.
[9] Jiaaho Chen. 2018. Fair lending needs explainable models for responsible recommendation. In FAT*EC’18 Proceedings of the Second Workshop on Responsible Recommendation. Vancouver, British Columbia, Canada, 4. arXiv:1809.06848.
[10] Jiaaho Chen, Nathan Kallius, Xiaojie Mao, Geoffrey Svacia, and Madeleine Udell. 2019. Fairness Under Uncertainty: Assessing Disparity When Protected Class Is Unknown. In Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT ’19). Association for Computing Machinery, New York, NY, 339–348.
[11] Alexandra Chouldechova. 2017. Fair Prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments. Big Data 5, 2 (2017), 153–163.
[12] Ethan Cohen-Cole. 2011. Credit card redlining. Rev. Econ. Stat. 93, 2 (2011), 708–713.
[13] Barbara A. Conway and Harry V. Roberts. 1983. Reverse Regression, Fairness, and Employment Discrimination. Journal of Business & Economic Statistics 1, 1 (1983), 75–85.
[14] Sam Corbett-Davies, Emma Pierson, Ari Feller, Sharad Goel, and Aziz Huq. 2017. Algorithmic Decision Making and the Cost of Fairness. In Proceedings of the 23rd

Table 2: Estimated coefficients \( \beta_1 \) and \( \beta_2 \) in the respective linear regression models \( M(1) \) and \( M(2) \), along with their standard errors \( (se) \), Wald statistic \( (6) \), and its \( p \)-value as determined from the \( \chi^2 \)-distribution. Bold \( p \)-values are significant at the \( \alpha = 0.05 \) level; values marked ** are significant at the Bonferroni-corrected \( \alpha/\nu = 0.0006 \) level. Only the Never-married and Widowed categories of marital status show statistically significant differences between \( \beta_1 \) and \( \beta_2 \).
Table 3: Estimated coefficients $\beta$ and $\zeta$ in the respective logistic regression models $M (3)$ and $M' (4)$, with other notation as defined in Table 2. None of the features are found to show statistically significant differences between the coefficients $\beta_i$ and $\zeta_i$.  

| Feature | $M(3)$ | $M'(4)$ | $\beta$ | $\zeta$ | 95% CI | 95% CI |
|---------|--------|--------|--------|--------|--------|--------|
| Married-all-spouse | 0.044 | 0.085 | (0.033, 0.055) | (0.074, 0.095) |
| Married-c-cw-absent | 0.039 | 0.080 | (0.028, 0.050) | (0.069, 0.090) |
| Married-cw-spouse | 0.049 | 0.088 | (0.038, 0.058) | (0.078, 0.098) |
| Never-married | 0.035 | 0.078 | (0.024, 0.045) | (0.067, 0.087) |
| Separated | 0.046 | 0.089 | (0.035, 0.057) | (0.074, 0.099) |
| Divorced | 0.054 | 0.092 | (0.043, 0.064) | (0.081, 0.103) |
| Asian-Hispanic | 0.040 | 0.082 | (0.029, 0.063) | (0.071, 0.092) |
| Craft-worker | 0.013 | 0.026 | (0.001, 0.027) | (0.013, 0.025) |
| Exec-managerial | 0.038 | 0.073 | (0.027, 0.049) | (0.062, 0.084) |
| Hours-per-week | 0.032 | 0.065 | (0.021, 0.043) | (0.054, 0.074) |
| High-school | 0.033 | 0.065 | (0.022, 0.044) | (0.053, 0.075) |
| Machine-op-inspect | 0.042 | 0.083 | (0.031, 0.053) | (0.072, 0.093) |
| Other-service | 0.046 | 0.089 | (0.035, 0.057) | (0.075, 0.098) |
| Peace-serv | 0.033 | 0.065 | (0.022, 0.044) | (0.053, 0.076) |
| Prof-specialty | 0.038 | 0.070 | (0.027, 0.049) | (0.062, 0.084) |
| Sales | 0.037 | 0.070 | (0.026, 0.048) | (0.059, 0.088) |
| Technical-support | 0.035 | 0.065 | (0.024, 0.045) | (0.054, 0.075) |
| Transport-moving | 0.035 | 0.070 | (0.024, 0.045) | (0.054, 0.075) |
| Other-occup | 0.122 | 0.225 | (0.110, 0.138) | (0.213, 0.237) |
| Other-prof | 0.125 | 0.230 | (0.112, 0.140) | (0.219, 0.241) |
| Other-white-collar | 0.121 | 0.228 | (0.109, 0.137) | (0.216, 0.240) |
| White-collar | 0.122 | 0.229 | (0.110, 0.140) | (0.217, 0.241) |
|也不敢 | 0.127 | 0.230 | (0.114, 0.140) | (0.218, 0.242) |

References:  
[19] Marshall Couchman, David Nebhut, and David Nickerson. 2000. Lessons Learned: Statistical Techniques and Fair Lending. Journal of Housing Research 11, 2 (2000), 277–295.  
[20] Julia Dressel and Harry Farid. 2018. The accuracy, fairness, and limits of predicting recidivism. Science Advances 4, 1. Article eaao5580 (2018), 5 pages.  
[21] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml  
[22] Olive Jean Dunn. 1961. Multiple Comparisons Among Means. J. Amer. Stat. Assoc. 56, 293 (1961), 52–64.  
[23] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidgen, and Suresh Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’15). Longing Cao, Chengqi Zhang, Thorsten Joachims, Geoff Webb, Dragos D Marculescu, and Graham Williams (Eds.). ACM, Sydney, NSW, Australia, 259–268.  
[24] Craig Gentry. 2009. Fully Homomorphic Encryption Using Ideal Lattices. In Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing (STOC ’09). Association for Computing Machinery, New York, NY, USA, 169–178.  

ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’17). Association for Computing Machinery, New York, NY, USA, 797–806.  
[25] Robert J. Serfling. 1980. Approximation Theorems of Mathematical Statistics. Wiley-Interscience, New York, Chapter 4, 138–170.  


[38] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. 2010. Fully Homomorphic Encryption over the Integers. In Advances in Cryptology – EUROCRYPT 2010. Henri Gilbert (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 24–43.

[39] Meredith Whittaker, Meryl Alper, Cynthia L. Bennett, and Emily Rogers. 2019. Disability, Bias, and AI. Technical Report. AI Now Institute at NYU, New York.

[40] Tal Zarsky. 2016. The Trouble with Algorithmic Decisions: An Analytic Road Map to Examine Efficiency and Fairness in Automated and Opaque Decision Making. Science Technology and Human Values 41, 1 (2016), 118–132.