Sara Jedwab

Professor Adam Kapelner Ph.D.

MATH 342W

March 21, 2021

Predict the Click on an Advertisement can be Modeled Well

These days the true currency is in people's attention, and endless resources are being allocated towards its capitalization; however, what exactly captures someone's attention is still not fully understood. Despite the fact that advertisements are incessantly popping up, humans have an innate ability to filter out useful information and block out the rest. Since half the battle is getting a consumer interested in one's product, the fittest companies are those that manage to create "click-bait" that is alluring enough to snag as many potential customers as possible, while the ones with advertisements that are ignored are unlikely to survive. So the potentially million-dollar question becomes: how can a company predict the likelihood of whether someone will click on their advertisement?

Promoting a product is a complex process that can take an entire marketing team to successfully pull off. There are many considerations that go into a person's decision - conscious and subconscious - whether or not to click on an advertisement. Identifying these psychological factors therefore becomes an integral part effectively publicizing - and thus selling - one's product, because once a company knows what the people want, they can give them just that. Fortunately, these days, technologies such as cookies allow websites to constantly collect data about the people that visit their site and potentially tailor their advertisements according to their preferences ("Online Tracking", 2018). So there are mass amounts of information being collected; yet in order to win that click, having the data alone isn't enough. A company must understand how to properly interpret the relevant data by creating a model which can predict the likelihood of whether a consumer will actually click on their advertisement.

From proverbial platitudes and intricate structures to abstracted systems, people are always creating "models" or representations which they can use to approximate their reality and the phenomena that they experience. Yet George Box, a British statistician, notably claimed that, "All models are wrong, but some are useful". In other words, from an epistemological perspective, "all models are wrong" by definition since they are only approximations which are not reality itself, and will therefore fall short of fully explaining the phenomenon. This is obviously true in our case: no one would claim to understand all possible factors that go into the "decision to click" and be able to predict with complete accuracy what a person will decide. Nevertheless, he asserts that "some [models] are useful" since sometimes they are good enough approximations to be used for a practical purposes. Generally those purposes can be split into two categories: prediction and explanation. When used for the former, a model can reveal what will happen with a certain phenomenon in a certain setting, and when used for the latter, a model can explain how reality really works or what causes a phenomena to manifest. In this case, the model will belong to the first category and be used to predict the likelihood of whether or not someone will click on a given advertisement.

In order to make the model as clear and unambiguous as possible, it is prudent to construct a *mathematical* model - one with numerical inputs and outputs - since it can capture the relationship between phenomena with an equation. Using numbers is a more objective way to create a model than using words since the latter are inherently open to a wide range of interpretations. Even so, mathematical models are still dependant on various assumptions; a critical one is *stationarity*, that a phenomena will remain the result of some (unknown) true causal drivers, and that the relationship between these true causal drivers and the features you've chosen to approximate the phenomenon will remain the same. In other words, stationarity is the assumption that the function used to model the phenomena doesn't change with time. If the function would change over time then after a certain amount of time elapsed the model would become obsolete and - more often than not - would not have been worth the effort of constructing.

Before constructing a mathematical model of a phenomenon it's very important to understand its structure. In reality, a phenomenon is the result of a certain amount of unknowable true casual drivers, represented as $z_1, z_2, ..., z_t$, and can thus be captured as a function: $y = t(z_1, z_2, ..., z_t)$. However, as mentioned above, all of these z_i's can never be truly known or fully quantified because they are realized in the future, and we also don't know the - potentially very complicated - function t that relates all these factors to one another to create the phenomena y. So the best that function can be created will be $y \approx f(x_1, x_2, ..., x_p)$, where each x_i is a proxy to approximate the information in the z_i's, and f is the best possible function given human limitations. There are p such proxies which are also referred to by a number of names such as: features, characteristics, attributes, variables, independent variables, covariates and more. Another way to write this function is $y = f(x_1, x_2, ..., x_p) + \delta$ such that $\delta = t - f$; where δ represents the error due to ignorance of the true casual drivers. There are various types of error that will be explored later on, but it's important to recognize that this type of error will *always* be present because the model will always be missing information because - at best - it still can only use proxies. Nevertheless, δ can obviously be reduced through learning more about the phenomenon and increasing the number of input features that more fully capture the true cause(s) of the phenomenon.

Due to the nature of this type of model, there is no analytical solution to determine the function f; instead, an empirical approach must be taken by learning from the data itself. There are various ways to to do this, but in this case it is best to employ *machine learning* from historical data. Machine learning is defined as "a branch of artificial intelligence (AI) focused on building applications that learn from data and improve their accuracy over time without being programmed to do so"(Education, n.d.). There are four main steps to this process: select and prepare a training data set, choose an algorithm to run on the training data set, training the algorithm to create the model and Using and improving the model (Education, n.d.). Additionally, we'll specifically be employing supervised machine learning, which, as opposed to unsupervised, utilizes a labeled dataset (Education, n.d.). So there

are essentially three components: training data ($\mathbb{D}$), a set of candidate functions ($\mathcal{H}$), and an algorithm ($\mathcal{A}$) that takes in $\mathbb{D}$ and $\mathcal{H}$ and returns the optimal approximation of f. The training data is comprised of n historical examples of inputs and outputs and is denoted $\mathbb{D} = \{< \vec{x}_1, y_1 >, < \vec{x}_2, y_2 >, ..., < \vec{x}_n, y_n >\}$. The set of candidate functions is comprised of elements h that approximate f; $\mathcal{H}$ is necessary because it limits the space of all possible functions which is too vast and nebulous to search through for the best function. Lastly, the algorithm that returns the best approximation of f will be denoted as $g = \mathcal{A}(\mathbb{D}, \mathcal{H})$. There will be a best function within the defined set of candidate functions, h^*, within $\mathcal{H}$ that ought to be used to model the phenomenon; how close g is to h^* will depend on which $\mathcal{A}$ is employed. It is also worth noting that typically $f \notin \mathcal{H}$, since f is arbitrarily complicated and unknown, and $\mathcal{H}$ usually contains only simple functions that can be fit with $\mathcal{A}$.

This methodology can be applied to create a mathematical model which can predict the likelihood of whether or not someone will click on an advertisement. In this case, the phenomenon being measured - denoted y - is a binary variable: an instance of an advertisement appearing on a computer screen but left untouched is assigned a 0, whereas an instance of it being clicked on is assigned a 1. The output space, $y \in \mathcal{Y} = \{0, 1\}$, is therefore discrete with a cardinality of 2. Since we collect this data with software, this metric is recorded faithfully without measurement error; however, there is no way to ascertain whether each click was intentional, so it is impossible to filter out accidental clicks. If there are enough accidental clicks in the data then this may impact the model, but at this point, we'll assume that all the observed clicks were intentional. There is another caveat that affects this model as a whole: this is not the only way to quantify effective advertising. It is possible that even if users don't click on an advertisement directly, they are still being influenced by its presence and may seek out the product at a different time (this metric is called "views" or "eyeballs" but we don't explore it herein).

Selecting the right type of modeling paradigm is essential for producing the best model to predict a phenomenon. Since this model has a binary output space, models like "Ordinary

Least Squares Regression" (OLS) and "K Nearest Neighnors" (KNN) are irrelevant here, and there are really only two types of models that we can construct: binary classification or probability estimation. The former could use an algorithm, $\mathcal{A}$, like the Support Vector Machine (SVM), that produces a line which maximizes the margins between the line and the data whilst minimizing the error or "hinge loss". This $\mathcal{A}$ would take in $\mathbb{D}$, $\mathcal{H} = \left\{\mathbb{1}_{\vec{w}\cdot\vec{x}-b\geq 0} : \vec{w} \in \mathbb{R}^p, b \in \mathbb{R}\right\}$, and a tuning hyper-parameter λ, and produce a g which, for each input x_*, would return $\hat{y}$, its predicted classification, either 0 or 1. This binary classification would be a simplistic prediction of whether or not someone will click on the advertisement. However, when carefully considering *why* we are building our model, it becomes evident that probability estimation is the significantly better model to build.

In our case, the model is being built for a company who wants to assess whether it is advantageous for them to use a particular advertisement on a given website. So a model which predicts the *likelihood* that someone will click on the advertisement provides much more information. In other words, a probability estimation model which, when given some x_*, predicts a 73.27% chance of a click, offers a lot more information than one which simply classifies that x_* as a click by producing a $\hat{y} = 1$, since you can compare probabilities in way that you can't compare zeros and ones. This type of mathematical model also allows us to factor in the price of advertising on the website and the profit the company stands to make from each sale, and perform an asymmetric cost analysis, which returns the optimal probability threshold. This is helpful because it could turn out that a 65% chance of a click is still not worth it, or it could turn out that that stand to gain even from a likelihood of 34%. Hence a probability estimation model best fits a company's needs in predicting the likelihood of a click on an advertisement, so how we construct it?

A probability estimation model is built in the following manner. Firstly, if $y \in \{0, 1\}$ for all i, then $y = t(\vec{z}) = f(\vec{x}) + \delta = h^*(\vec{x}) + \mathcal{E} = g(\vec{x}) + e$ where $\delta, \mathcal{E}, e \in \{0, -1, +1\}$. Then we now view Y as a realization from a random variable: $Y \sim \text{Bern}\big(t(\vec{z})\big)$. We will also assume ther exists a function $f_{pr}(\vec{x}) : \mathbb{R}^{p+1} \to (0, 1)$, which is the best guess of the

probability you can create with $\vec{x}$, $P(Y=1|\vec{x})$. So $Y \sim \text{Bern}\big(f_{pr}(\vec{x}) + t(\vec{z}) - f_{pr}(\vec{x})\big)$ where $t(\vec{z}) - f_{pr}(\vec{x}) = \delta_{pr} \implies Y \sim \text{Bern}\big(f_{pr}\big)$, thus f_{pr} is the model we want to find. If we assume that all the n observations which comprise $\mathbb{D}$ are independently realized then: $P(\mathbb{D} = P(Y_1 = y_1, Y_2 = y_2, ..., Y_n = y_n|\vec{x}_1, ..., \vec{x}_n) = \prod_{i=1}^{n} P(Y_i = y_i|\vec{x}_i) = \prod_{i=1}^{n} f_{pr}(\vec{x}_i)^{y_i}\big(1 - f_{pr}(\vec{x}_i)\big)^{1-y_i}$. Though we intend on learning from data in order to construct our model, we cannot fit f_{pr} using our data since it is impossible to fit arbitrary functions in any dimension. Instead we must define a set of candidate functions that we can fit, called $\mathcal{H}_{pr}$, where each element in this set maps $\mathbb{R}^{p+1} \to (0,1)$. Though it would simple to define $\mathcal{H}_{pr} = \{\vec{w} \cdot \vec{x} : \vec{w} \in \mathbb{R}^{p+1}\}$, this would return values out the space of legal probabilities which is $(0,1)$. So we employ a "link function" which takes $\vec{w} \cdot \vec{x}$ and maps it onto the space $(0,1)$: $\phi : \mathbb{R} \to (0,1)$, and we restrict it to be strictly increasing. Thus the true set of candidate functions for this model is: $\mathcal{H}_{pr} = \{\phi(\vec{w} \cdot \vec{x}) : \vec{w} \in \mathbb{R}^{p+1}\}$. These types of models are referred to as "generalized linear models" (glm) because they are just a manipulation of the linear model, $\vec{w} \cdot \vec{x}$.

There are three common types of link functions (in order of use): Logistic/Logit, Probit, and Complementary Log-Log/Cloglog. The logistic link function typically the best choice, and would probably perform best when modeling the likelihood of a click on an advertisement. The logit function is: $\phi(u) := \frac{1}{1+e^{-u}}$, thus the set of candidate functions becomes $\mathcal{H} = \big\{\frac{1}{1+e^{-\vec{w}\cdot\vec{x}}} : \vec{w} \in \mathbb{R}^{p+1}\big\}$. Hence the algorithm which takes in this $\mathcal{H}$ and the $\mathbb{D}$ described above is $\mathcal{A} : \vec{b} := \text{argmax}_{\vec{W} \in \mathbb{R}^{p+1}} \Big\{ \prod_{i=1}^{n} \big(\frac{1}{1+e^{-\vec{w}\cdot\vec{x}}}\big)^{y_i} \big(\frac{1}{1+e^{\vec{w}\cdot\vec{x}}}\big)^{1-y_i} \Big\}$, which essentially just finds the $\vec{w}$ that provides the highest probability based on $\mathbb{D}$. As opposed to other algorithms, there is no analytical solution in this case. All you can do is use a computer to set $\vec{\nabla} P(\mathbb{D}) = \vec{0}_{p+1}$ and approximate. This is usually done with "gradient descent" or running a logistic regression which computes $\vec{b}$. After this, we can (finally) predict using $\hat{p}(Y=1|\vec{x})\hat{p} = g_{pr}(\vec{x}) = \phi(\vec{b} \cdot \vec{x}) = \frac{1}{1+e^{-\vec{b}\cdot\vec{x}}}$, which will hopefully be close to f_{pr}. The slope coefficients (i.e. the $\vec{b}$ entries) can be understood as b_j being the change in the log-odds of $Y=1$ if x_j increases by 1 (these are obviously highly non-linear).

After determining the proper modeling paradigm, we must construct the dataset. As

mentioned above, the z_i's and the t function that produce the click phenomenon are unknown, so they are approximated with the p features (x_i's) and f respectively. Nevertheless, it is imperative to first thoroughly consider what the casual drivers *could* be, in order to then determine the optimal features for a model. In our case the casual drivers could include the user's desire and/or need for the advertised product, whether the user has the funds to purchase the advertised product, how much time the user typically spends perusing products online, the attractiveness of the ad, whether the user recognizes/trusts the product's company, and the caliber of the website that on which the advertisement appears. This list is obviously incomplete and hypothetical, and even these potential z_i's are ambiguous and impossible to measure exactly.

Thus, in order to preserve the integrity of our model's predictions, it is necessary to pick the best features which will serve as proxies for the casual drivers. This demands thorough knowledge of both the phenomenon and its (potential) z_i's which is usually gleaned from extensive research. With these ideals in mind and given that features will be limited by quantifiability and availability, the best general features for this model are therefore: gender, age, socioeconomic standing, daily time spent on website, daily internet usage, appearance of ad, attention capture and website quality ("Predicting Customer Ad Clicks via Machine Learning", n.d.; Richardson et al., 2007). Again, it is important to recognize that we are assuming stationarity with our model: firstly, that the click on an advertisement will remain the result of the same z_i's (hypothetically including the ones enumerated above and the others that were omitted due to ignorance), and secondly, that the relationship between these true causal drivers and the features we've chosen to approximate them will remain the same. This assumption is necessary since our model will really only be useful if we can assume that the function used to model the phenomena doesn't change with time. Yet it is not only necessary, but reasonable to assume that the psychology behind clicking on an advertisement is unlikely to change, at least in the near future.

The first variable of this model is gender which is a nominal feature broken down into

three categories: male, female and non-binary. It is dummified by using non-binary as the reference category, and so it is represented in two columns of the X matrix. Though this categorization is not all-encompassing of the wide spectrum of gender identification, it is broad enough to capture what is going on without adding unnecessary features (which, as will be discussed later, could lead to overfitting). One could argue that this is not a good feature since advertisements are much less gendered then they used to be due to a breakdown of classical gender roles, and that there is movement towards more gender neutral advertising. Nevertheless, it is evident that marketing teams still target specific genders when selling certain products by analyzing psychological differences between them (Chisholm, 2013). So assessing the gender of the user will be useful in predicting whether or not they will click on a given advertisement.

Additionally, age is a relevant feature to include when constructing the dataset for this model. This a discrete variable, measured in years, definitely contributes to "the click" as certain products are definitely geared towards specific age groups, and their advertisements will be tailored accordingly. For example, an advertisement for college preparatory resources are going to likely be targeting a younger crowd, and so the age of the user will factor into the model of the likelihood s/he will click on that advertisement.

Next, the socioeconomic standing of the user is likely to influence whether they will click on a given advertisement. This will be a continuous variable measured by determining the mean income of the area in which the search is being conducted. A user in a "wealthier" area (measured in this way), will likely have more expendable income to spend and may be more likely to click on an advertisement (and vice versa with more impoverished areas). Additionally - though this is more difficult to quantify - if an advertisement is targeting people of a specific socioeconomic standing, it will effect the prediction whether users are within that range.

Furthermore, both daily time spent on website and daily internet usage are continuous features that will impact the accuracy of the model's prediction. They are measured in

minutes per the classical 24-hour day from 12 am to 11:59 pm within the user's timezone. It might seem redundant to include daily time spent on website since that is kind of already included in daily internet usage; however, both of those will independently contribute to the model. It makes sense to claim that whether someone spends a lot of time online will contribute to the likelihood that they click on an advertisement; those with more time to spend on the internet are presumed to be more likely to click on an ad. This however neglects a subsection of people who, though they don't spend a lot of time online relative to other users, their time spent online is mainly on this specific website. When that is the case, it makes sense that this should influence the prediction, because this could account for low daily internet users still producing clicks on an advertisement. It is therefore worth including both of these features in this model.

The features above are focused on the features pertaining the users themselves, but it's also important to take into consideration aspects of the advertisement itself. These aspects can be broken down into two general categories: appearance of the ad, the ad's attention capture. These features were suggested by Microsoft researchers Richardson, Ragno, and a Dominowska, when researching a similar type of model; they also offered ways to define more precise metrics for these obviously very ambiguous variables.

The appearance of the ad essentially answers the question: "is the ad aesthetically pleasing?" (Richardson et al., 2007) It makes sense to assume that people will be drawn towards an advertisement that is more aesthetically appealing, and thus more likely to click on it. However, this is still a broad question must be broken down in smaller questions like: "How many words are in the title? In the body? Does the advertisement have good capitalization? Does it contain too many exclamation points, dollar signs, or other punctuation? Does it use short words or long words?" (Richardson et al., 2007) Each of these are more concrete and can be translated into quantifiable features that can measure the ad's appearance. Thus the features in this category are as follows. Firstly, there are number of words in the title and number of words in the body; it's likely that there is an optimal number of words which is

pleasing to the eye, not too many or too few. Then "good capitalization" will be assessed by counting up the number of errors which violate the grammar rules; it makes sense to suppose that a person could be put off by an grammar error so blatant as improper capitalization. Additionally, there will be features which count the number of exclamation points, dollar signs and question marks; when used in moderation, these symbols could likely make the ad more appealing. Lastly, there will be two more features, one which counts how many short words are in the ad and another to count the long ones; anything that is six letters or under will be short, and more than that will be considered long. Again, this may influence the ad's overall attraction. All of these features will be discrete variables where each value is an integer greater than or equal to zero. Thus this category has eight features in total.

The next category, attention capture, is very similar to the first, but its more focused on answering: "does the ad draw the user in?" (Richardson et al., 2007) This also ought to be broken down into questions like,"Does the title contain action words such as "buy", "join", "subscribe", etc.? Does the body? Does the ad provide numbers (such as specific discounts, prices, etc)?" (Richardson et al., 2007) In this category, the first features will again be discrete variables with values greater than or equal to zero; the first and second features are counts of how many actions words there are in the title and body respectively. It's sensible to presume that action words might subconsciously influence a person to do that action or want to do that action. The third will be a binary variable whether or not the ad has numbers in it. This is worth including because maybe seeing a specific price, discount, quantity etc. will impact the user's interest. Another relevant binary feature here is whether or not there is a sound that comes on when the ad appears on the screen; a noise is likely to capture the user's attention to the ad and increase the chance of a click. Both of these latter binary variables will be measured as 0 when absent and 1 when present. Hence attention capture category is divided into four features.

Lastly, it's important to also take into consideration the quality of the website displaying the advertisement. This in it of itself requires complex analysis (and can be a project in

it of itself), but for the purposes of this model, it can be broken down into the following features. The first is popularity of the website measured in number of users visiting the site on the day of the observation. The more popular the site, the higher the probability that the advertisement will get clicked on. The second is a count of how many advertisements appear on the website; if there are a lot of other competing ads then it's probably less likely that the company's ad will be clicked on. Third is page loading speed measured in seconds; it's possible that the longer the site takes to load the less willing the user will be to click on an ad which will navigate him/her away from the site, forcing him/her to reload it to get back ("How to asses the quality of website", 2020). There are obviously many other factors which go into the quality of a website (like its aesthetics), but are (at least at this juncture) more difficult to boil down into quantifiable metrics. So at this point, there are three features within this category.

After getting into the specifics of these features and their values to the model as a whole, it is worth discussing why some seemingly relevant features were been left off the list. Firstly, the IP address of the user was not included even though observations are quantified based on an instance of the advertisement on a user's screen, so there can be multiple observations with the same user. It also makes sense that whether someone has clicked on a similar advertisement in the past would be relevant in predicting the likelihood that they will click on the one now before them. Though this may or may not be true, at this time it is too difficult to quantify this in our model because this would be a nominal feature that, when dummified, would add too many columns to the X matrix (and again, probably lead to overfitting). Similarly, it for this reason that the "topic" of the advertisement wasn't included as feature, despite the fact that it might be useful. This cost is too great to be worth the inclusion of these features, though it is worth acknowledging that theoretically they would likely be contributing factors to the phenomenon. Additionally, it is likely that the preexisting reputation of the company being advertised will effect the likelihood that the ad will be clicked on, but this is a very difficult feature to quantify with precise metrics.

The features that were selected to comprise $\mathbb{D}$ however, are collected from various sources. The data for the first four features about the user - gender, age, socioeconomic standing, daily time spent on website, daily internet usage - will be accumulated from the moment that the user visits a specific site. A limitation of this dataset will then obviously be those who deny the website's request to use cookies. The data for the features within the categories of the ad's appearance and attention capture will be collected from the company using the model to test our their various advertisements. Lastly, the data about the website quality will be acquired from the website hosting the advertisement; the data for features in this category are not too difficult for the website to determine and not too private, and they are likely to volunteer this information if they hope to continue business with the company in question. Overall, it is safe to assume that measuring these features will be practical and that there would be more than enough historical data to provide a sample to fit the model. Also, though its possible (probable) that the dataset will be missing some values, the values that are recorded should have high accuracy.

The information for these features are recorded in a row vector $x_i = [x_{i1}, x_{i2}, ..., x_{ip}]$ and the collection of these x_i vectors comprise the X matrix. Furthermore, whether or not the person clicks on the ad is recorded as y_i, and are all recorded in a single column vector $\vec{y} = [y_1, y_2, ..., y_n]$. An observation is recorded for each instance of an ad that appears on a person's screen (so there could be multiple observations attributed to one person); each observation is denoted as $< \vec{x}_i, y_i >$. All of the n observations - which are just the combination of X and $\vec{y}$ - comprise the dataset $\mathbb{D}$ for this model. Additionally, since there are 21 features, there will be $p = 21$ columns. However, if analysis of interactions between features is relevant, then p would increase with every interaction included.

Furthermore, after we create our model, we need to ensure that it is in fact a good predictive power. To assess the level of its performance, it's important to clarify the error metric that's generally used for a probability estimation model. There is a clear challenge here since the true probabilities f_{pr} are never observed, and it is difficult to compare the

labels (0 or 1) to the probabilities produced by our model. To solve this problem we use an error metric called a "scoring rule" S that can compare a $\hat{p}$ value to a y value. A "proper scoring rule" $S(\hat{p}, y)$ is one where: $\forall_i f_{pr}(\vec{x}_i) = \text{argmax} = \{S(\hat{p}_i, y_i)\}$. There are two main types of proper scoring rules: the Brier Score and the Log Scoring Rule. The former was developed in 1950 and is a bit more common; it is defined as $s_i := -(y_i - \hat{p}_i)^2$. The latter is defined as $s_i := y_i \ln(\hat{p}_i) + (1 - y_i) \ln(1 - \hat{p}_i)$. Both of these scores will always be less than or equal to zero (as will each of their averages), and the closer the score is to zero, the better the probability estimation model. So, though the exact numerical threshold for usefulness is impossible to ascertain at the theoretical level, it is definitely true that a useful model will have a Brier Score and Log Scoring rule with values closer to zero. Also, it must at least beat the Brier score of the simplest model which is $g_0 = \bar{y}$.

Nevertheless, error scores themselves may be misleading when the model is either under-fit or over-fit. "Under-fitting occurs when a statistical model cannot adequately capture the underlying structure of the data" ("Overfitting", 2021). This can be the result of not including interactions between features in the dataset or limiting the candidate set to functions that are not complex enough, which will lead to a model that is too simple to capture the relationship between the features that produce phenomenon. Even just not having enough features could lead to under-fitting, since the model will fail to capture enough of what's going on. Conversely, over-fitting is "the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably" ("Overfitting", 2021). This is typically a problem when trying to fit an overly complicated model like with higher-degree polynomials, when the number of features, p approaches the total number of observations, n, or if there are features present which don't substantially contribute to the phenomenon. This is why the process of selecting x_i's is so important; since, despite being a bit non-intuitive, the addition of any feature will always contribute to the overall fit of the model in some way and make the error value go down. This is deceptive because in reality over-fitting increases "gener-

alization error" - the model's inability to generalize to future predictions; this is really bad because the whole point of creating the model is to produce accurate predictions!

However, the deception is exposed through a process called validation, which more accurately assesses the performance of g. This is done by first randomly splitting the dataset into two groups D_{train} and D_{test} where the latter is typically 10% - 20% of $\mathbb{D}$. This proportion is usually denoted by specifying a K, where $\frac{1}{K} :=$ proportion in the test set, so the defaults are therefore $K = 5$ or 10. D_{train} is what is included in $\mathcal{A}$ when actually generating the model, whereas after we already have g, D_{test} is used in calculating metrics which give allow an honest assessment of the model's performance. This allows us to calculate two type of error metrics: in-sample and out-of-sample. Whereas in-sample metrics are calculated using the same data that trained the model, oos error metrics are computed using only the "testing data" (as will be described in more detail below). This oos error metrics - by assuming stationarity - inform us of the model's predictive accuracy in the future because the model is being tested with data it's never analyzed. So if the model too closely fits the training data that it cannot generalize, then, though the in sample error metrics will look fantastically low, oos error metrics will sky-rocket. However, often times simple validation is not enough since the oos error metrics can vary based on the specific random split, and are themselves random variables. If their variance is high, then our estimates aren't so useful. So we reduce this variance by performing "K-fold Cross Validation" which essentially makes K splits and repeats the train-test validation procedure. Each observation in the original dataset gets represented once inside of a test set. Each split "crosses over" the test set to the next set of $\frac{1}{k} * n$ indices. This reduces variance because we are averaging many realizations of the oos error random variable. Another bonus is that we can also compute oos metrics in each of the K splits, which enables you to gauge the variability of the oos errors. Though it is true that for this to be valid, the error metrics would have to be independent (which they are not because they use a lot of the same data), this still gives some degree of guarantee of the oos estimate. Furthermore, selecting a large K versus a small one will effect the oos

estimate of performance. The main trade-off is that if K is large, then n_{test} will be small and thus the oos estimate of performance will be highly variable because its an estimate with very little data, but since n_{train} will be almost n it will not be as biased, and the converse is true as well. In our model it is thus prudent to split our dataset and perform K-fold Cross Validation (with the usual $K = 5$ or 10) and assess its performance using both oos Brier Scores and Log Scores.

Though our click prediction model could obviously be over-fit - which is typically the more common issue whenever modeling - in this case, our model is more vulnerable to under-fitting. It's unlikely that our model would be over-fit because the features selected seem to each substantially contribute to the phenomenon. Furthermore, the n observations are anticipated to far surpass the mere $p = 21$ features. However, there is a concern that the model will be under-fit, because firstly the dataset described previously does not include an any interactions where even though there are probably interactions going on. For example, age might interact with socioeconomic status, or time spent on website with total internet usage. Including interactions would allow for our model to better capture more complex relationships between features, and as it stands, the model might fall short. Additionally, the psychology behind a person's decision to click is complex, and its possible that there are many more features that could to be included that would produce a better fit model.

Now that we've got $\mathbb{D}$, and the $\mathcal{H}$ and $\mathcal{A}$ defined above, and we know how to assess its performance, we can finally create the model g_{pr}! Given the same p raw features, there are, firstly, many transformations and interactions where one can augment the design matrix, and secondly, many different algorithms. So you can be many different models. Since all models for real phenomena are only "approximate", there is no "correct" model, but we still need to choose one. This is the fundamental issue of model selection, which essentially just picks the one with lowest oos error; the procedure is as follows. First the dataset is divided into $\mathbb{D}_{train}$ and $\mathbb{D}_{test}$, then $\mathbb{D}_{train}$ is divided again into $\mathbb{D}_{subtrain}$ and $\mathbb{D}_{select}$ where the latter is $\frac{1}{k}$ of the data within $\mathbb{D}_{train}$. All (let's say M) models are built using the $\mathbb{D}_{subtrain}$, then the

errors of all M models are assessed using $\mathbb{D}_{select}$. ($\mathbb{D}_{select}$ acts as the $\mathbb{D}_{test}$ for model selection only). Finally, K-fold split crossing is performed to assess the error of each of the M models, and the one with the best performance will be picked as m_*. Then fitting m_* on all $\mathbb{D}_{train}$, we can assess its oos performance using $\mathbb{D}_{test}$. Yet still the oos estimate from the one $\mathbb{D}_{test}$ can be highly variable, so its best to cross-validate that as well in a process called "nested re-sampling". Again, picking K_{test} and K_{select} will have the same bias-variance trade-off considerations but for different decisions. In each of the outer re-samplings we locate a different m_*, and final oos error will be on the meta-algorithm ($\mathcal{A}$) that selects the best of the M models. Lastly, g_{final} is computed by running the model selection procedure on all of $\mathbb{D}$ which will pick one $m_* = g_{final}$. In the case of our model, after performing this procedure, it's likely that the selected probability estimation model will use logit as its link function (most common), and will have the lowest oos Brier and/or oos Log scores.

Nevertheless, no matter how well the model performs, it will always fall short primarily due to three types of error: estimation error, model misspecification error, and error due to ignorance. This can be denoted as:

$$y = g(\vec{x}) + \underbrace{\underbrace{h^*(\vec{x}) - g(\vec{x})}_{\text{estimation error}} + \underbrace{\underbrace{f(\vec{x}) - h^*(\vec{x})}_{\text{(model misspecification error)}} + \underbrace{t(\vec{z}) - f(\vec{x})}_{\delta\text{ (error due to ignorance)}}}_{\varepsilon}}_{e\text{ (residual error)}}$$

Error due to ignorance is caused by our lack of knowledge about what we're studying and of all the true drivers that lead to the output(s); this error can be reduced by doing research and learning more about what we're trying to model, but it can never be fully eliminated. With our model, this error will probably be high because it is very difficult to fully capture the psychology behind what impels a person to click on an advertisement. Also, it's possible that some of the features aren't as relevant as they were hypothesized to be, or that some important features were left out because they were too difficult to quantify (like reputation of company being advertising). *Model misspecification error* happens when the model doesn't fit the data well enough; this error can be reduced by expanding the set of candidate functions

$\mathcal{H}$ to be more complicated and thus more expressive of complex relationships. In the case of our model, this error will probably also be on the higher end because (as mentioned above) it hasn't taken any interactions into consideration, where there likely are some. *Estimation error* occurs when the model cannot accurately predict outputs because it doesn't have enough training data; this error can be reduced by increasing sample size (more historical examples/the rows in $\mathbb{D}$). Thankfully, with our model, this error should be relatively low since it shouldn't be difficult to obtain large amounts of data in the ways described previously.

Besides for the limitations of the accuracy of the model's prediction due to these errors, the model will also be limited by the range of values it can predict. It's essential to limit use of our model to interpolation which is predicting for x-vectors inside the range of X.That means limiting our model to only predict on values that were already used to train it. In this case the x values must be within the dataset's ranges of gender classifications, ages, incomes, time spent on website, internet usage and so on and so forth for the rest of the 21 features. The opposite of this is called extrapolation, which is incredibly dangerous because different model fitting procedures ($\mathcal{A}$) extrapolate differently. It also just makes sense that the model won't be able to accurately provide predictions about values that it never evaluated. So for example, if there is no data on anyone 75 or older, than the model cannot predict accurately whether an 83 year old will click on the ad. Similarly, if all the observations in the dataset have at least 8 words in the body, then the model cannot produce an accurate prediction on how a user will react to an add with 3 words in the body. Therefore, in order to best utilize the model, predictions should be limited to interpolation, and avoid extrapolation at all costs.

Building a model to predict the likelihood that a user will click on an advertisement is useful venture that can provide useful analysis to many companies. Fortunately, as outline above, it is definitely possible to model this phenomenon, and model it well. With the very clear features outlined above and the wealth of data that can made readily available, this model could be built with relatively high predictive accuracy.

Works Cited

Chisholm, S. (2013). Gender and advertising how gender shapes meaning.

Education, I. C. (n.d.). What is machine learning? https://www.ibm.com/cloud/learn/machine-learning

How to asses the quality of website. (2020). https://snapics.co/how-to-asses-the-quality-of-website/

Online tracking. (2018). https://www.consumer.ftc.gov/articles/0042-online-tracking#:~:text=When%20you%20visit%20a%20website,your%20device%20in%20the%20future.&text=By%20keeping%20track%20of%20you,deliver%20ads%20targeted%20to%20you.

Overfitting. (2021). https://en.wikipedia.org/wiki/Overfitting#cite_note-1

Predicting customer ad clicks via machine learning. (n.d.). https://stackabuse.com/predicting-customer-ad-clicks-via-machine-learning/

Richardson, M., Dominowska, E., & Ragno, R. (2007). Predicting clicks: Estimating the click-through rate for new ads. *Proceedings of the 16th international conference on World Wide Web - WWW 07*. https://doi.org/10.1145/1242572.1242643