Alexander Montes McNeil

NUID: 001989922

Nishanth Marer Prabhu

NUID: 002624650

EECE5644 Spring 2023

Final Project

For the EECE 5644 final project we chose to analyze a data set of all NFL games outcomes since the year 2000. Specifically, we started with the preprocessed generated from this github repository: <https://github.com/ukritw/nflprediction> . Our goal was to see if an artificial neural network could approach the Empirical Risk Minimization (ERM) estimate for this data to predict the outcome of each game. This is binary classification problem where the labels for this dataset are 1 if the home team wins or 0 if the away teams wins. For each game the dataset contains a list of attributes about the home and away team such as the team name, their winning percentage over the current and past season, and the fivethirtyeight elo percent chance of winning for each side. In total the dataset has over thirty features so the first step was to understand the features and eventually remove any unnecessary attributes. We used a correlation matrix of all the numerical attributes of the data better understand it. A below we display a histogram of all the numerical features in the dataset with the result of the game.



As expected, the result has a 1:1 correlation with the result of the game itself. The rest of the numerical data features have a value of between 1 and -1. A feature is closely related with the result of the games (‘result’ in the plot above) if the absolute value of the correlation coefficient is close to 1. We chose to keep features that were +/- 0.05 from qualitative analysis on the correlation results above. Some features such as the home/away team scores cannot be included in our decision because it is the result of the game itself. Ultimately, we chose to keep the following features for our classification problem based on the plot above: elo\_prob1/2, team\_home/away\_current\_win\_pct, team\_home/away\_lastseason\_win\_pct, home/away\_favorite, and spread\_favorite.

Before we could perform the ERM, we had to perform some manipulation of the dataset. Since this is a real dataset, we had to determine the class posteriors for the outcome of the game to be label 0 (the away team wins) or label 1 (the home team wins). Then we could determine the mean and covariance values of the data for each class. We should note that, to ensure the covariance matrix was invertible, we had to regularize it.

Now we can evaluate the data from each label with it’s pdf and find the likelihood that is a member of each class. We use the following algorithm to achieve this.

1. Compute *:*
2. Compute the product of the class priors with the class conditional pdf :
3. Divide by to obtain the class posterior value :
4. Determine the risk matrix:
5. Take the argmin of the risk matrix to determine which class a sample belongs to.

The confusion matrix for this algorithm applied to our dataset is shown below.

|  |  |
| --- | --- |
| ERM Confusion Matrix | |
| 0.50 | 0.23 |
| 0.50 | 0.77 |

We used the neural network model from the python scikit-learn package to build our neural network model. We were interested in how the effect of the number of folds used during cross-validation would affect the overall performance of the dataset. We tried 2, 5, 10, 15, and 20 folds on the dataset to determine the optimal number of neurons per layer. We split our dataset into training data and test data with 20% of our dataset going to training data and the last 80% going to test data. Then we preformed the k-fold cross validation with our training data to determine the optimal number of neurons, from 1 to 20. The results of our models are shown below.

Chart, line chart

Description automatically generated

The MLP clearly outperforms the ERM estimate for this data (Note: that the ERM and MAP estimates are equivalent for a 0-1 loss matrix). We estimate that this could be due to the shape of the dataset. Additionally, our dataset is relatively small with 4738 samples. We propose two improvements for the future. The first is to expand the dataset to games before the year two thousand. The second is to better understand the shape of the data, for example we will determine if it follows a Gaussian distribution.

All data and code can be found in our github repository for this project located here: <https://github.com/nishanthmarer/ITMLPRFinalProjectAlex_Nishanth>