**Curtis Bell**

**DATA SCIENCE**

Email: Curtis.Bell18@outlook.com

**SUMMARY**

* **Data Scientist with 6+ years of experience** in Machine Learning, Deep Learning, Natural language processing, Computer Vision, Image Processing and Quality Analyst in the domain of E-commerce, Retail and Software Consulting.
* Implemented **Machine Learning**, **Computer Vision**, **Deep Learning** and **Neural Networks** algorithms using **TensorFlow**, **Keras** and designed Prediction Model using Data Mining Techniques with help of Python, and Libraries like **NumPy, SciPy, Matplotlib, Pandas, Scikit-learn.**
* Well versed with common NLP Techniques such as **tokenization, part-of-speech tagging, parsing, stemming, named entity recognition, semantic analysis,** **Modeling and word representations (TF-IDF, LSA, LDA, word2vec)** using libraries **NTLK, Spacy, Gensim, Polyglot, TextBlob.**
* Experienced with **RStudio, Jupyter Notebooks, LSTM** and **RNN algorithms.**
* Experience in performing **Feature Selection**, **Feature Engineering**, **Linear Regression, Logistic Regression, k-Means Clustering,** **Hierarchical (Agglomerative) clustering, Classification, Decision Tree, Support Vector Machines (SVM), Naive Bayes, K-Nearest Neighbors (KNN), Random Forest, and Gradient Descent,** **Neural Network algorithms** to train and test the data sets.
* Used **pandas, NumPy, Seaborn, SciPy, matplotlib, Scikit-learn, NLTK** in Python for developing various machine learning algorithms.
* Applied **Support vector machines (SVM)** and it's **kernels** such **Polynomial, RBF-kernel** on machine learning problems.
* Worked with deep neural networks **and Convolutional Neural Networks (CNN's)** and **Recurrent Neural networks (RNN's**)
* Implemented Classification using supervised algorithms like **Logistic Regression, SVM, Decision trees, KNN, Naive Bayes**
* Evaluated models using **Cross Validation**, **Log loss function, ROC curves** and used **AUC** for feature selection and elastic technologies like **Elastic Search, Kibana etc**.
* Addressed **overfitting** by implementing the algorithm **regularization** methods like **L2 and L1** and **dropouts** in neural networks.
* Implemented **statistical modeling** with **XGBoost** machine learning software package using Python to determine the predicted probabilities of each model.
* Worked with different performance metrics like **log-loss, AUC, confusion matrix, f1-score** for **classification** and **mean square error, mean absolute error** for regression problems.
* Formulated a basis for variable selection and **Grid Search, KFold** for optimal **hyperparameters**.
* Used **Principal Component Analysis** and **T-SNE** in feature engineering to analyze high **dimensional data.**
* Performed **Data Cleaning, features scaling, features engineering** using **pandas and NumPy** packages in python and build models using deep learning frameworks.
* Implemented application of various machine learning algorithms and statistical modeling’s like **Decision Tree, Text Analytics, Sentiment Analysis, Naive Bayes, Logistic Regression and Linear Regression** using Python to determine the accuracy rate of each model.
* Strong familiarity in working with various statistical concepts such as **A/B Tests, Hypothesis Testing, t-Test, and Chi - Square Test, ANOVA, Statistical Process Control, Control Charts, Descriptive Statistics and Correlation Techniques.**
* Application of various machine learning algorithms and statistical Modeling like **decision trees, text analytics**, **natural language processing (NLP),** **supervised and unsupervised.**
* Experienced in creating databases, populating it, to extract data from data tables along with creation of tables, **sub queries, joins, views, indexes, SQL functions** and other functionalities.
* Proficient in Microsoft Word, Advanced Excel – (**VLOOKUP, Macros, Pivot Table**), PowerPoint.
* Implemented **CBOW, Skip Grams, Bag of Words, Named Entity Recognition**, **POS Tagging.**
* Performed **data transformations** using **log, square-root, reciprocal, differencing or a complete box-cox** transformation depending upon the dataset.
* Expertise **in developing time-series forecasting models** such as Auto-regressive & integrated moving averages (ARIMA), multiplicative & additive decomposition, exponential smoothing and winters multiplicative model to predict values exhibiting seasonal or cyclical patterns

**SKILLS**

|  |  |
| --- | --- |
| **Analytical Techniques** | Hypothesis Testing: Cohen’s d and t-test, Independent & pairwise t-tests, one-way and two-way factorial anova; Regression Methods: Linear, polynomial, poisson, decision trees, Support vector; Classification : Logistic Regression, K-NN, Decision Trees, Naïve Bayes, Support Vector Machines (SVM), Clustering: K-means clustering, Hierarchical Clustering; Deep Learning: Artificial Neural Networks, Convolutional Neural Networks; Dimensionality Reduction: Principal Component Analysis (PCA); Linear Discriminant Analysis (LDA); Text Mining: Natural Lnaguage Processing (NLP),; Ensemble learning : Random Forest, Gradient Boosting Machine, Bagging |
| **Algorithms** | Gradient descent, Stochastic Gradient descent |
| **Validation Techniques** | Gradient, Monte-Carlo Simulations, K-Fold Cross Validation, out-of-bag sample estimates |
| **Data Visualization Tools** | Tableau, Microsoft Power BI, ggplot2 and plotly, Python matplotlib, seaborn, bokeh |
| **Data modeling** | Entity Relationship Diagrams (ERD), Snowflake schema, Star Schema |
| **Languages** | PostgreSQL, Microsoft SQL server, R, Python, Octave (MATLAB), Hadoop |
| **Database Systems** | SQL Server 10.0/11.0/13.0, Oracle, MYSQL 5.1/5.6/5.7, Teradata, DB2, Amazon Redshift, SAP Hana |
| **NoSQL Databases** | HBASE, Apache Cassandra |
| **ETL Tools** | KAFKA, FLUME, Microsoft SSIS |
| **Big Data** | Apache, Hadoop, HDFS, Sqoop, Flume, Hive, Impala, MapReduce, Splunk, ML-SPL, Splunk Hadoop Connect, OOzie |

**WORK EXPERIENCE**

**Walmart** Bentonville, AR (01/2019)-Present

*Data Scientist/Computer Vision Engineer*

*Description:*

Trained an image classifier with **VGGNET** as network architecture including Convolution layers, Dropout layers, Fully Connected layers and improved accuracy by 16% over this model with InceptionResNetV2 model as part of Transfer learning.

*Roles/Responsibilities:*

* Imported packages: **NumPy, Pandas, keras, os, open cv, matplotlib, seaborn, random, gc, sklearn**
* **Reshaped** the images to the size relevant as the input to the **neural network**
* Confirmed graphically the **distribution of classes** in the **training dataset**
* Used **sklearn train\_test\_split** to split the training dataset **to 80% as training dataset and 20% as validation dataset.**
* Imported modules -**layers, models, optimizers, ImageDataGenerator, img\_to\_array, load\_img** - from keras.
* **Normalized** input image pixel values to have a unit standard deviation and mean of 0.
* Chosen the **VGGNET** as **the network architecture**.
* Created a **Sequential** model with **Conv2D** layer having **filter size 32, kernel size of 3,3 relu as activation function, and input shape of 150, 150, 3**. Also, added **pooling layer-MaxPool2D layer, Flatten layer, Dropout layer, sigmoid as activation function** for the output layer.
* Used **binary\_crossentropy** as a **loss function** for the classification problem
* Specified **acc** as **accuracy metric** and **rmsprop** as **optimizer** as part of **hyperparameter tuning**.
* Created **ImageDataGenerator** object for training and test dataset.
* Fitted the model using **model.fit.generator** function specifying **train\_generator, steps per epoch, number of epochs, validation generator and validation steps**
* Achieved **80% accuracy** with **64 epochs** on **validation dataset**.
* Plotted **loss vs epoch and accuracy vs epoch** and concluded that on increasing the epoch size will likely give higher accuracy and loss will likely go down.
* Imported **InceptionResNetV2** for **Transfer learning** with **weights** trained on **ImageNet** dataset without downloading the **fully connected layers** of the pretrained model from keras.
* Created a model with first layer as InceptionResNetV2, Flatten layer, Dense layer with relu activation function, another Dense layer as output layer with sigmoid as activation function.
* Used **RMSprop** with **learning rate** of **0.0002** during **model compilation** and metric as acc
* Achieved **accuracy of 96% in 20 epochs**.
* Deployed the model inproduction environment **(AWS Cloud)** using **Dockers and Kubernetes.**

**Client: Medtronic(Health Care), Los Angeles, CA (01/2016)- (12/2018)**

**Role: Data Scientist/Machine Learning Engineer**

Responsibilities:

• Predicted patient’s BG and insulin metrics based on the monthly data

• Used Time Series analysis to forecast future BG values

• Used Pandas to read and load TS data

• Converted object datatypes into time series objects

• Evaluated the stationarity of the time series data

• Used Scikit learn to clean data and to categorize into train and test data sets

• Worked with Tensorflow using Python to construct the model

• Programmed and executed code and algorithms for applications

• Used NumPy and Pandas libraries for data computation and extraction

• Used Input functions to feed data when training and evaluating the model

**Zappos,** Las Vegas, NV (01/2013)- (12/2015)

*Deep Learning Engineer*

*Description:*

Built a **recommendation engine** with **probabilistic matrix factorization mode**l in **PyTorch.**

*Roles/Responsibilities:*

* Performed **exploratory data analysis (EDA)** on the data set using **pandas**.
* **Normalized** **rating** values and examined the **sparsity of the data** set or the number of known ratings in comparison to number of possible user item combinations.
* **Filtered the dataset** to include only user that had rated at least k times.
* Implemented a **recommender** model in **PyTorch** using the **Embedding class** in **PyTorch**, which creates a 2-dimensional **embedding matrix**.
* Added **bias term** to the model by creating two embedding layers – one for the user bias and one for the item bias.
* After filtering and **pre-processing**, determined the first **hyperparameter** - **learning** **rate** - by **training** the **model** for several **iterations**, and choosing the learning rate for which model **loss** is **minimized**.
* Determined the other hyperparameter- **latent dimensionality** using **grid search algorithm** and choosing the **embedding size** for which **validation loss** is **minimal**.
* **Trained** the model for **3 epochs** with **learning rates of .1, .05, .01, .005, and .001**, resulting with a **final MSE loss of 0.75.**