Script

Slide 1:[Introduction]

Hello Everyone. My topic for the presentation is the development of a Cryptocurrency Price Prediction Platform. So let's get into it.

Slide 2 :[Problem Statement]

So let's first look into the problem statement and the methods that we needed to inculcate in our methodology.
We start our discussion on Data preprocessing on our given dataset and analyze which pre-processing techniques are well suited along with feature engineering.
We'll then discuss our regression models, their suitability, advantages and limitation pertaining to our given task. Finally, we'll take into consideration the various evaluation metrics through which are models are scored on their accuracy and error rates.

Slide 3:[Data Preprocessing]

So what is data preprocessing? Data preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis. The large dataset is bound to have either redundant data or would have missing values or a number of probable outliers which can hinder our results during training and evaluation.
So let's look into some of the methods we pre-process data.
First is handling of Missing Data: This means that we check for missing values in each column of the dataset. Once identified, we can treat them either by removing the entire row or by filling the empty cell with the appropriate value by calculating the mean or median.

Secondly, we can remove irrelevant columns and features by identifying which of these would not be very useful for our training process For eg. ID, Slug, Num_market_pairs, Max_supply along with platform-related features can easily be removed if they are considered irrelevant for price prediction.
Thirdly, we clean up the dataset by removing duplicates and handling outliers. Identifying outliers basically means treating those values that are very distant from real-time values and should not be taken into consideration in raw form during the training process.

Next is data normalization which means to scale features like *Market_cap, Volume_24h, Circulating_supply, and Total_supply* to a similar scale.
And finally, if there are categorical features like Names or Symbols, it's better to encode them using techniques like one-hot encoding or label encoding to convert them into numerical values.

Slide 4:[Feature Engineering]

Moving on to feature engineering which actually is a part of the pre-processing, however, due to it's vital contribution to the accuracy of the model, I am discussing this in a separate slide. This process means transforming raw data into features that can be used for creating a predictive model using machine learning. Here I've listed five most relevant feature engineering method which contribute most to our model.

First up is Date-based Features: If we look into our dataset we'd find these two columns namely *'date_added' and 'last_updated*'. These can be used to extract useful features such as the day of the week, month, or year. These are helpful in determining the potential seasonality or periodic patterns in cryptocurrency prices.
Second, we consider the factor of historical price features where we create new features that represent the percentage change in price over different time intervals. The reason these features are helpful is that they can capture short-term price trends and volatility easily.
Thirdly, Rolling Statistics is the method to find averages of price and volume over a specific time window. This is done to capture trends and smooth out short-term fluctuations.
Fourth is employing lagged features: So For example, you can create features representing the price from previous time steps (e.g., the previous day's price) to capture dependencies and momentum effects. These are very useful when working with a time series forecasting problem.

Seasonality is one of the very useful feature engineering methods yet not much talked about. The crux of this feature is that if there are known seasonal or holiday effects on cryptocurrency prices, we can create features that capture such patterns. For example, if there is a historical trend of increased trading activity or price movement during Christmas, we can include a binary or categorical feature to represent those periods. This will help us in countering any abnormalities witnessed in crypto prices on those particular days.

Before moving on, I must address that it's important to consider the limitations of historical data and the potential for changing market dynamics when incorporating engineered features.

Slide 5:[Machine Learning Models]

Coming to the core of our prediction system, which is choosing the best four models for predicting accurate cryptocurrency prices for future dates. I've shortlisted the following four models based on various parameters that we're going to discuss.

Slide 5 & 6[Random Forest Regressor]

Let's understand the Random Forest algorithm by the given diagram. Let's consider our crypto price dataset for example. What RF would do is it'll split our dataset based on some given parameters. The split dataset is then fed into an individual decision tree. And once these

decision trees have given an output of the defined parameter result, we average them out based on majority voting. Its pretty simple after that. The one with the highest weightage or vote for the resulting prediction will be chosen and considered as our future price.
Now that we have an overview of our model let's look into some aspects.

Now the question in your head would be why and how this model is suitable for crypto price prediction. One of the reasons is that it handles non-linearity (often seen in crypto/stocks), making it suitable for capturing the intricate dynamics of cryptocurrency prices.
Secondly, if you remember we talked about the problem of outliers in our pre-processing slide. The good thing is that the RF Regressor is robust to outliers as it uses an ensemble of decision trees, which reduces the impact of individual outliers on the overall model. So even if you haven't considered treating outliers, the Random Forests have got you covered!

Moreover, RF Regressor provides a measure of feature importance. This basically means understanding which features have a stronger influence on the predicted prices.
And Finally, for a large dataset (like ours) the Random Forest can handle high-dimensional data efficiently without requiring extensive feature selection or dimensionality reduction techniques. And this of course saves us time and computation!

One handy thing about RF is that it mitigates overfitting by either introducing random samples or randomly selecting features each time we split. If you don't know what overfitting is, it's basically an undesirable factor in ML and is seen when models give highly accurate metric on the training data but lose all their might on the test or new data. For eg. if overfitting is not treated, we might not be able to predict the crypto prices for newer values, which is basically the whole purpose of our task.
The preprocessing tasks our the ones listed and were mostly discussed in the previous slides. For training the data, normally we take the split to be 80:20.
RF Regressor has been widely used in various domains, including finance and stock market prediction, which share similarities with cryptocurrency price prediction.
At last limitations of this model can include overfitting on the dataset provided along with being computationally expensive compared to other models.

Slide 7:[Extra Trees Regressor]

Similar to Random Forests, ExtraTrees is another tree-based ensemble ML approach. As mentioned, it was introduced by Pierre Geurts, Damien Ernst, and Louis Wehenkel in 2006. So the question arises how is it different from the RF model?

Well, RF uses bagging meaning individual datapoint chosen more than once to select different variations of the training data to ensure decision trees are sufficiently different. **However**, Extra Trees uses the entire dataset to train decision trees.
If you would look at the attached diagram you'd find that the splits are randomized for every decision tree in Extra Trees. Much of the later process remains the same up till the final result.

The reason why we employ this model can now be understood from the time we talked about RF regressors. They handle non-linear data well, no pressure of preprocessing outliers, it can pick up defined important features to have a bias towards them. But also, in terms of computational cost, Extra Trees is much faster than Random Forest. This is because Extra Trees randomly selects the value at which to split features, instead of the greedy algorithm used in Random Forest.

Moving on to applications of extra tree regressor which has been successfully applied.
For eg. in financal time series analysis (like in our case) can be used to predict stock prices, exchange rates, and commodity prices based on historical price data and other relevant features. Also in Credit Risk Assessment, Extra Trees Regressor has been utilized to predict credit risk by analyzing customer financial data, credit history, and other relevant factors.

Talking about the Limitations of Extra Trees, the**y** tend to have higher variance compared to Random Forest models due to the additional randomness introduced during feature selection and split point selection. By variance, I mean changes in the model when using different parts of the training data. High variance is hence not something that we would like to have in our model. This leads us to do manual research and analytics on deciding which features should be marked important for the model, which is of course, time-consuming.

Slide 9:[Facebook’s Prophet Model]

Prophet is an open-source tool from Facebook used for forecasting time series data which helps businesses understand and possibly predict the market. It is based on a decomposable additive model where non-linear trends fit with seasonality, it also takes into account the effects of holidays.
Prophet is specifically designed for time series forecasting, making it well-suited for predicting future cryptocurrency prices based on historical patterns.
Cryptocurrency markets often exhibit seasonal patterns, periodic fluctuations, and long-term trends. Prophet incorporates mechanisms to capture such seasonality and trends, allowing it to model and predict these patterns accurately. Prophet is also designed to be user-friendly and accessible, even for users without extensive expertise in time series forecasting.

Stepping to our next Algorithm,

To explain the crux algorithm of the model and why is it suitable, let’s consider an example.
Imagine you're running a small bakery, and you want to forecast your daily sales for the next month. You have historical data of your daily sales for the past year, and you want to use Facebook's Prophet to help you with the predictions.
First, Prophet's trend model will analyze the historical sales data and identify the overall trend. It will figure out if your sales have been steadily increasing or decreasing. Let's say it finds that your sales have been increasing gradually.
Next, the Prophet has a mode as a seasonality model. It basically looks for any patterns that repeat at fixed intervals. For example, it might detect that your sales tend to be higher on

weekends and lower on weekdays. It will also capture any monthly or yearly cycles, such as increased sales during holidays or special events like Christmas or New Year.
Now, let's say you have a holiday approaching next month. Prophet allows you to input this information as a custom event. It will take into account that this particular holiday typically brings in more customers and adjusts the forecast accordingly.
Prophet is also clever enough to handle missing data. If there are days when you didn't record sales or if you have some outliers in your data, Prophet can handle them gracefully. It can fill in the gaps and still provide reliable predictions.
Combining all these components, Prophet generates a forecast for your daily sales for the next month. It gives you an estimate of how much you can expect to sell each day, considering the overall trend.

One interesting thing about the Prophet is that Prophet requires minimal preprocessing as most of it is inbuilt into the model. However some preprocessing, if needed, like handling non-stationary data to increase the accuracy of the model.
Applications of Prophet are mainly seen in demand forecasting, web traffic prediction, etc.

At last, looking into the limitations of the model Prophet assumes that the underlying trends and seasonality follow linear patterns. This means it may not capture complex nonlinear relationships in the data. Secondly, Prophet focuses primarily on modelling the time series components (trend, seasonality) and may not handle external factors well. As a result, it may not be suitable for scenarios where the target variable is influenced by a significant number of external factors beyond the provided data.

Slide 11: [Lasso Regression Model]

Lasso stands for "Least Absolute Shrinkage and Selection Operator.", is a linear regression model that was introduced by Robert Tibshirani in 1996 as a modification of traditional linear regression.
Before diving into the Lasso model, it's important to know of a term known as **Regularization**. Remember when we talked about overfitting which is when a model does not perform good on test data? Regularization is the method to counter overfitting. And our Lasso is just that, a regularization technique! This technique can be used in such a way that it will allow to maintain all variables or features in the model by reducing the magnitude of the variables. Hence, it maintains accuracy as well as a generalization of the model.

Lasso regression works like a detective trying to solve a mystery. It wants to find the hidden clues that strongly influence the prices. It starts by assigning a weight to each feature in your dataset. The weight represents how much impact a particular feature has on the price.

Consider the feature called "Date_Added." Lasso regression will analyze the data and assign a weight to this feature. A higher weight means that it has a stronger influence on the price, while a lower weight means it has less impact.

Now, here's where Lasso regression gets interesting. It wants to simplify the model and only keep the most important features. So, it starts penalizing the weights of less important features by shrinking them towards zero.
In our example, let's say Lasso regression finds that the ‘num_market_pairs’ and the ‘ID’ have relatively low impacts on crypto prices compared to the volume_change. It will shrink the weights of these features closer to zero, indicating their reduced significance.
Why does Lasso regression do this? Well, by shrinking the weights towards zero, it effectively removes the less important features from the equation. This simplification helps us focus on the most influential features, making our predictions more accurate and easier to interpret.

For eg. look at the figure attached. The basic idea of lasso regression is to introduce a little bias so that the variance can be substantially reduced, which leads to a lower overall MSE(Mean Squared Error). Notice that as λ increases, variance drops substantially with very little increase in bias. Beyond a certain point, though, variance decreases less rapidly and the shrinkage in the coefficients causes them to be significantly underestimated which results in a large increase in bias.
And When λ = 0, the penalty term in lasso regression has no effect and thus it produces the same coefficient estimates as least squares.

Talking about applications, The Lasso Regressor has been successfully applied in various domains, including finance, economics, and healthcare.
If talking about limitations, Lasso Regressor tends to select features based on their individual predictive power and may overlook important feature interactions. Moreover, it consistently selects the same features for a given dataset and regularization parameter. This can be a limitation if the dataset characteristics change or if different subsets of features are relevant under different circumstances.

In comparison with both the RF and Extra Tree(Tree-based models) we conclude that by making a number of features disappear, Lasso or L1 is a more simple and interpretable model.
Moreover, Tree-Based models perform the feature selection process at each split. However, RF and ET are better at handling non linear relationships. And finally L1 fares better to both RF and ET when talking about computation effeciency.
This marks the end of our model analysis.

Slide 13:[Evaluation Metrics]

Now to measure how well our model fared on the dataset it’s important to measure them using some predefined metrics: R-squared for example measures the proportion of variance in the target variable that is explained by the model. It ranges from 0 to 1, where 1 indicates a perfect fit and 0 indicates that the model does not explain any of the variances. Second, Mean Squared

Error(MSE) calculates the average squared difference between the predicted values and the actual values. In totality, lower MSE values indicate better performance. In Root Mean Squared Error (RMSE) we find out the square root of MSE and provide a more interpretable metric in the same unit as the target variable.
And finally, Mean Absolute Error (MAE) calculates the average absolute difference between the predicted values and the actual values. On a similar note, lower MAE values indicate better performance.

Slide 14:[Conclusion]

In conclusion, we have discussed the development of a Cryptocurrency Price Prediction Platform, focusing on the dataset provided. We started off by exploring the importance of data preprocessing.
Furthermore, we discussed four machine learning models suitable for predicting cryptocurrency prices: Random Forest Regressor, Extra Trees Regressor, Facebook's Prophet Model, and Lasso Regression Model. We highlighted the advantages and limitations of each model and provided examples of their applications in various domains.
We conclude that the Random Forest Regressor is well-suited for handling non-linearity, robustness to outliers, and high-dimensional data. The Extra Trees Regressor offers faster computation and has been successfully applied in finance and predictive analytics. Facebook's Prophet Model is designed specifically for time series forecasting, capturing seasonality and trends, while Lasso Regression Model performs variable selection and regularization well.
It is important to note that each model has its own limitations, such as overfitting in the Random Forest Regressor, higher variance in the Extra Trees Regressor, linearity assumption in Facebook's Prophet Model, and limited handling of external factors in the Lasso Regression Model.
In conclusion, the choice of the best out of the four models for the Cryptocurrency Price Prediction Platform depends on the specific requirements and characteristics of the dataset. Careful consideration should be given to the features, data patterns, computational resources, and interpretability needed for accurate cryptocurrency price prediction.

Slide 15:[Conclusion]

That’s all from my side. Thank you, and questions are welcome